Note:
Before using this information and the product it supports, read the information in "Notices" on page H-1.
Contents

Introduction ........................................... xv
In This Introduction ................................... xv
About This Publication ................................. xv
  Types of Users ...................................... xv
  Software Dependencies .............................. xvi
  Assumptions About Your Locale ................... xvi
  Demonstration Database ............................. xvi
New Features .......................................... xvi
Documentation Conventions ............................ xix
  Typographical Conventions ......................... xix
  Feature, Product, and Platform Markup .......... xx
  Example Code Conventions ......................... xx
Additional Documentation ............................. xxi
Compliance with Industry Standards ................. xxi
Syntax Diagrams ....................................... xxi
  How to Read a Command-Line Syntax Diagram .. xxii
  Keywords and Punctuation ......................... xxiii
  Identifiers and Names ............................. xxiii
IBM Welcomes Your Comments ....................... xxiv

Part 1. Configuring and Monitoring Dynamic Server

Chapter 1. Configuration Parameters .................. 1-1
In This Chapter ......................................... 1-4
ONCONFIG File Conventions ............................ 1-4
  Format of ONCONFIG File ......................... 1-4
  ONCONFIG File Templates ......................... 1-4
  Printing the onconfig.std File ................... 1-5
  Specifying Hidden Configuration Parameters ..... 1-5
  Displaying ONCONFIG Settings .................... 1-5
Summary of Configuration Parameters ............... 1-6
  Parameter Attributes ............................... 1-11
  Using a Utility to Change a Parameter Value ... 1-11
  Environment Variables ............................. 1-12
Archecker Configuration Parameters .................. 1-12
ADMIN_MODE_USERS .................................. 1-13
ADMIN_USER_MODE_WITH_DBSA ....................... 1-13
ADTERR, ADTMODE, ADTPATH, and ADTSIZE (UNIX) .... 1-14
ALARMPROGRAM ....................................... 1-14
ALLOW_NEWLINE ....................................... 1-14
ALRM_ALL_EVENTS ..................................... 1-15
AUTO_AIOVPS ......................................... 1-15
AUTO_CKPTS .......................................... 1-16
AUTO_LRU_TUNING .................................... 1-16
AUTO_REPREPAIR ...................................... 1-16
BLOCKTIMEOUT ....................................... 1-17
BTSCANNER ........................................... 1-17
BUFFERPOOL .......................................... 1-18
  The Iru Field ....................................... 1-18
  The buffers Field ................................... 1-18
  The lru_min_dirty Field ......................... 1-19
  The lru_max_dirty Field ......................... 1-20
  The size Field .................................... 1-20
  System Page Size ................................. 1-22
CKPTINTVL ........................................... 1-22
<table>
<thead>
<tr>
<th>Parameters</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEANERS.</td>
<td>1-23</td>
</tr>
<tr>
<td>CONSOLE</td>
<td>1-24</td>
</tr>
<tr>
<td>DATASKIP</td>
<td>1-24</td>
</tr>
<tr>
<td>DBCREATE_PERMISSION</td>
<td>1-25</td>
</tr>
<tr>
<td>DB_LIBRARY_PATH</td>
<td>1-25</td>
</tr>
<tr>
<td>DBSERVERALIASES</td>
<td>1-26</td>
</tr>
<tr>
<td>DBSERVERNAME</td>
<td>1-27</td>
</tr>
<tr>
<td>DBSPACETEMP</td>
<td>1-27</td>
</tr>
<tr>
<td>Using Hash Join Overflow and DBSPACETEMP</td>
<td>1-29</td>
</tr>
<tr>
<td>DD_HASHMAX</td>
<td>1-30</td>
</tr>
<tr>
<td>DD_HASHSIZE</td>
<td>1-30</td>
</tr>
<tr>
<td>DEADLOCK_TIMEOUT</td>
<td>1-30</td>
</tr>
<tr>
<td>DEF_TABLE_LOCKMODE</td>
<td>1-31</td>
</tr>
<tr>
<td>DIRECT_IO (UNIX)</td>
<td>1-32</td>
</tr>
<tr>
<td>DIRECTIVES</td>
<td>1-32</td>
</tr>
<tr>
<td>DISABLE_B162428_XA_FIX</td>
<td>1-33</td>
</tr>
<tr>
<td>DRDA_COMMBUFFSIZE</td>
<td>1-33</td>
</tr>
<tr>
<td>DRAUTO</td>
<td>1-34</td>
</tr>
<tr>
<td>DRIDXAUTO</td>
<td>1-35</td>
</tr>
<tr>
<td>DRIPTERMAL</td>
<td>1-35</td>
</tr>
<tr>
<td>DRLOSTFOUND</td>
<td>1-35</td>
</tr>
<tr>
<td>DRTIMEOUT</td>
<td>1-36</td>
</tr>
<tr>
<td>DS_HASHSIZE</td>
<td>1-36</td>
</tr>
<tr>
<td>DS_MAX_QUERIES</td>
<td>1-37</td>
</tr>
<tr>
<td>DS_MAX_SCAN</td>
<td>1-37</td>
</tr>
<tr>
<td>DS_NONPDQ_QUERY_MEM</td>
<td>1-38</td>
</tr>
<tr>
<td>DS_POOLSIZE</td>
<td>1-39</td>
</tr>
<tr>
<td>DS_TOTAL_MEMORY</td>
<td>1-39</td>
</tr>
<tr>
<td>Algorithm for DS_TOTAL_MEMORY</td>
<td>1-40</td>
</tr>
<tr>
<td>DUMPCNT (UNIX)</td>
<td>1-40</td>
</tr>
<tr>
<td>DUMPCORE (UNIX)</td>
<td>1-41</td>
</tr>
<tr>
<td>DUMPPDIR</td>
<td>1-41</td>
</tr>
<tr>
<td>DUMPGCORE (UNIX)</td>
<td>1-42</td>
</tr>
<tr>
<td>DUMPSHMEM (UNIX)</td>
<td>1-42</td>
</tr>
<tr>
<td>DYNAMIC_LOGS</td>
<td>1-43</td>
</tr>
<tr>
<td>ENCRYPT_CIPHERS</td>
<td>1-43</td>
</tr>
<tr>
<td>ENCRYPT_HDR</td>
<td>1-45</td>
</tr>
<tr>
<td>ENCRYPT_MAC</td>
<td>1-46</td>
</tr>
<tr>
<td>ENCRYPT_MACFILE</td>
<td>1-46</td>
</tr>
<tr>
<td>ENCRYPT_SMX</td>
<td>1-47</td>
</tr>
<tr>
<td>ENCRYPT_SWITCH</td>
<td>1-48</td>
</tr>
<tr>
<td>Enterprise Replication Configuration Parameters</td>
<td>1-49</td>
</tr>
<tr>
<td>EXPLAIN_STAT</td>
<td>1-50</td>
</tr>
<tr>
<td>EXT_DIRECTIVES</td>
<td>1-50</td>
</tr>
<tr>
<td>EXTSHMADD</td>
<td>1-51</td>
</tr>
<tr>
<td>FASTPOLL</td>
<td>1-51</td>
</tr>
<tr>
<td>FILLFACTOR</td>
<td>1-51</td>
</tr>
<tr>
<td>HETERO_COMMIT</td>
<td>1-52</td>
</tr>
<tr>
<td>IFX_EXTEND_ROLE</td>
<td>1-52</td>
</tr>
<tr>
<td>IFX_FOLDVIEW</td>
<td>1-52</td>
</tr>
<tr>
<td>ISM_DATA_POOL and ISM_LOG_POOL</td>
<td>1-53</td>
</tr>
<tr>
<td>Java Configuration Parameters</td>
<td>1-53</td>
</tr>
<tr>
<td>LISTEN_TIMEOUT</td>
<td>1-54</td>
</tr>
<tr>
<td>LOCKS</td>
<td>1-54</td>
</tr>
<tr>
<td>LOGBUFF</td>
<td>1-55</td>
</tr>
<tr>
<td>LOGFILES</td>
<td>1-56</td>
</tr>
<tr>
<td>LOG_INDEX_BUILDS</td>
<td>1-56</td>
</tr>
<tr>
<td>LOGSIZE</td>
<td>1-57</td>
</tr>
<tr>
<td>LOGSIZE for Smart Large Objects</td>
<td>1-57</td>
</tr>
<tr>
<td>LTAPEBLK</td>
<td>1-58</td>
</tr>
<tr>
<td>LTAPEDEV</td>
<td>1-58</td>
</tr>
</tbody>
</table>
LTAPESIZE .......................................................... 1-59
LTXEHWM .......................................................... 1-59
LTXHWM .......................................................... 1-60
MAX_FILL_DATA_PAGES ........................................ 1-61
MAX_INCOMPLETE_CONNECTIONS .............................. 1-62
MAX_PDQPRIORITY .............................................. 1-62
MaxConnect Configuration Parameters ...................... 1-63
MIRROR .......................................................... 1-63
MIRROROFFSET .................................................. 1-64
MIRRORPATH ..................................................... 1-64
Setting Permissions (UNIX) ..................................... 1-65
MSGPATH ........................................................ 1-65
MULTIPROCESSOR .............................................. 1-65
NETTYPE ........................................................ 1-66
   Protocol ....................................................... 1-67
   Number of Poll Threads .................................... 1-68
   Number of Connections .................................... 1-68
   Class of Virtual Processor ................................. 1-68
   Default Values .............................................. 1-69
   Multiplexed Connections .................................. 1-69
   IBM Informix MaxConnect ................................. 1-69
OFF_RECVRY_THREADS ......................................... 1-69
ON_RECVRY_THREADS .......................................... 1-70
ON-Bar Configuration Parameters ............................ 1-70
ONDBSPACEDOWN ............................................... 1-71
   Database Server Behavior When ONDBSPACEDOWN Does Not Apply .................................... 1-71
ONLIDX_MAXMEM ............................................... 1-72
OPCACHEMAX (UNIX) ............................................ 1-72
OPTCOMPIND ................................................... 1-73
OPT_GOAL ....................................................... 1-73
PC_HASHSIZE ................................................... 1-74
PC_POOLSIZE ................................................... 1-75
PHYSBUFF ....................................................... 1-75
PHYSDBS ........................................................ 1-76
PHYSFILE ........................................................ 1-76
QSTATS .......................................................... 1-77
PLOG_OVERFLOW_PATH .......................................... 1-77
RA_PAGES ........................................................ 1-77
RA_THRESHOLD .................................................. 1-78
RESIDENT ....................................................... 1-78
RESTARTABLE_RESTORE ....................................... 1-79
ROOTNAME ...................................................... 1-80
ROOTOFFSET ................................................... 1-80
ROOTPATH ....................................................... 1-81
ROOTSIZE ....................................................... 1-81
RTO_SERVER_RESTART .......................................... 1-82
SBSPCENAME ................................................... 1-82
SBSPCETEMP ................................................... 1-83
SDS_ENABLE .................................................... 1-84
SDS_PAGING .................................................... 1-84
SDS_TEMPDBS .................................................. 1-84
SDS_TIMEOUT .................................................. 1-85
SECURITY_LOCAL_CONNECTION ............................... 1-86
SERVERNUM ..................................................... 1-86
SHMADD ........................................................ 1-86
SHMBASE ....................................................... 1-87
SHMTOTAL ....................................................... 1-88
SHMVIRT_ALLOCSEG ......................................... 1-88
SHMVIRTSIZE .................................................. 1-89
SINGLE_CPU_VP ............................................... 1-90
   User-Defined VP Classes and SINGLE_CPU_VP .......... 1-91
Chapter 2. The sysmaster Database

In This Chapter
2-2
The sysmaster Database
2-2
The buildsmi Script
2-2
The bldutil.sh Script
2-3
The System-Monitoring Interface
2-3
Understanding the SMI Tables
2-3
Accessing SMI Tables
2-4
The System-Monitoring Interface Tables
2-4
The sysutils Tables
2-6
sysadinfo
2-6
sysaudit
2-7
sysckio
2-7
syscheckpoint
2-8
syschunks
2-8
sysconfig
2-9
sysdatabases
2-10
sysdblocale
2-10
sysdbspaces
2-10
sysdri
2-11
sysdual
2-12
sysenv
2-12
sysenvs
2-12
sysextent
2-12
sysextspaces
2-13
syslh
2-13
Chapter 3. The sysadmin Database ....................................................... 3-1
In This Chapter ................................................................. 3-1
The sysadmin Database ........................................................ 3-1
The PH_TASK Table .......................................................... 3-1
The PH_RUN Table .............................................................. 3-3
The PH_GROUP Table .......................................................... 3-3
The PH_ALERT Table ........................................................... 3-3
The PH_THRESHOLD Table ....................................................... 3-5
The Results Table ............................................................... 3-5
The command_history Table .................................................. 3-5

Chapter 4. Disk Structures and Storage ...................................... 4-1
In This Chapter ................................................................. 4-2
Dbspace Structure and Storage .................................................. 4-2
  Structure of the Root Dbspaces ............................................. 4-2
  Reserved Pages ............................................................. 4-3
  Structure of a Regular Dbspaces .......................................... 4-3
  Structure of the Chunk Free-List Page ................................. 4-4
  Structure of the Tblspace Tblspace ..................................... 4-5
  Structure of the Database Tblspace .................................... 4-7
  Structure and Allocation of an Extent ................................ 4-8
  Structure and Storage of a Dbspaces Page ............................. 4-12
  Structure of Fragmented Tables ........................................... 4-15
  Structure of B-Tree Index Pages ......................................... 4-16
  Structure of R-Tree Index Pages .......................................... 4-21
Storage of Simple Large Objects ............................................. 4-21
  Structure of a Blobspace .................................................. 4-21
  Structure of a Dbspaces Blobpage ....................................... 4-21
  Simple-Large-Object Storage and the Descriptor ................. 4-22
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
</table>

**Chapter 5. Interpreting Logical-Log Records**

In This Chapter ........................................................................................................ 5-1

**About Logical-Log Records** ........................................................................... 5-1

- Transactions That Drop a Table or Index ........................................... 5-1
- Transactions That Are Rolled Back .................................................... 5-2
- Checkpoints with Active Transactions ............................................. 5-2
- Distributed Transactions ........................................................................ 5-2

**Logical-Log Record Structure** .................................................................... 5-3

- Logical-Log Record Header ..................................................................... 5-3
- Logical-Log Record Types and Additional Columns ....................................... 5-3
- Log Record Types for Smart Large Objects ........................................... 5-14

**Part 2. Administrative Utilities**

**Chapter 6. Overview of Utilities** .................................................................. 6-1

In This Chapter ........................................................................................................ 6-1

**Complete List of Utilities** ........................................................................ 6-1

- Obtaining Utility Version Information ........................................... 6-2
- Multibyte Characters (GLS) ................................................................. 6-2

**OpenAdmin Tool for IDS** ............................................................................... 6-2

**IBM Informix Server Administrator** ............................................................ 6-3

**Server Studio** ................................................................................................... 6-4

**Chapter 7. The oncheck Utility** .................................................................. 7-1

In This Chapter ........................................................................................................ 7-1

**oncheck Check-and-Repair** ....................................................................... 7-1

- What Does Each Option Do? ............................................................... 7-2
- Using the -y Option to Perform Repairs ........................................... 7-3
- Repairing Indexes in Sbspaces and External Spaces .............................. 7-3
- Locking and oncheck ........................................................................ 7-3

**Oncheck Syntax** ............................................................................................. 7-4

- oncheck -cc and -pc: Check system catalog tables .......................... 7-8
- oncheck -cd and -CD: Check pages ................................................... 7-9
- oncheck -ce, -pe: Check the chunk-free list ..................................... 7-10
- oncheck -ci and -CL: Check index node links ................................... 7-10
- oncheck -cr and -CR: Check reserved pages ...................................... 7-12
- oncheck -cs, -cs, -ps, -pS: Check and display sbspaces ....................... 7-12
- oncheck -pB: Display blobspace statistics ........................................ 7-13
- oncheck -pd and pD: Display rows in hexadecimal format ............... 7-13
- oncheck -pk, -pK, -pl, -PL: Display index information ................... 7-15
- oncheck -pp and -pp: Display the contents of a logical page ................... 7-16
- oncheck -pr and -pR: Display reserved-page information .................. 7-18
- oncheck -pt and -pt: Display tbspaces for a table or fragment .......... 7-18
- Turn On Locking with -x .................................................................... 7-20
- Send Special Arguments to the Access Method with -u ..................... 7-20
- Return Codes on Exit ........................................................................ 7-20

**Chapter 8. The ondblog Utility** ................................................................. 8-1
In This Chapter ................................................................. 8-1
ondblog: Change Logging Mode ............................................ 8-1
   ondblog Syntax ......................................................... 8-1

Chapter 9. The oninit Utility .............................................. 9-1
oninit: Initialize the Database Server ................................... 9-1
   oninit Syntax .................................................................. 9-2
   Initialize Shared Memory Only ........................................ 9-2
   Initialize Disk Space and Shared Memory ........................... 9-3
   Specify the Number of Virtual Processors ........................... 9-3

Chapter 10. The onlog Utility ............................................... 10-1
onlog: Display Logical-Log Contents ..................................... 10-1
   onlog Syntax .................................................................. 10-2
   Log-Record Read Filters ................................................ 10-2
   Log-Record Display Filters ............................................ 10-3

Chapter 11. The onmode Utility ............................................. 11-1
In This Chapter .................................................................. 11-1
onmode Syntax .................................................................. 11-3
   onmode -a: Add a shared-memory segment ......................... 11-4
   onmode -BC: Allow large chunk mode ................................ 11-4
   onmode -c: Force a checkpoint ......................................... 11-5
   onmode -C: Control the B-tree scanner ............................... 11-6
   onmode -d: Set data-replication types ................................. 11-7
      Using the -d standard Option ...................................... 11-7
      Using the -d primary dbservname Option ....................... 11-7
      Using the -d secondary dbservname Option ................. 11-7
   onmode -d: Set secondary server characteristics ................. 11-8
   onmode -d: Replicate an index with data-replication .......... 11-9
   onmode -D, -M, -Q, -S: Change decision-support parameters 11-10
   onmode -e: Change usage of the SQL statement cache ........ 11-11
   onmode -F: Free unused memory segments ......................... 11-12
   onmode -k, -m, -s, -u, -j: Change database server mode ...... 11-12
      Taking the Database Server to Offline Mode with the -k Option 11-13
      Bringing the Database Server Online with the -m Option .... 11-14
      Shutting Down the Database Server Gracefully with the -s Option 11-14
      Shutting Down the Database Server Immediately with the -u Option 11-14
      Changing the Database Server to Administration Mode with the -j Option 11-14
      Changing Database Server Mode with ON-Monitor (UNIX) .... 11-15
   onmode -l: Switch the logical-log file ............................... 11-15
   onmode -n, -r: Change shared-memory residency ................. 11-15
   onmode -O: Override ONDBSPACEDOWN WAIT mode ............. 11-16
   onmode -p: Add or remove virtual processors ..................... 11-17
      Adding and Dropping Virtual Processors ....................... 11-18
      Dropping Virtual Processors Automatically ................... 11-18
      Monitoring Poll Threads with onstat ............................ 11-19
   onmode -R: Regenerate .infos File .................................. 11-19
   onmode -W: Change settings for the SQL statement cache .... 11-20
      SQL Statement Cache Examples .................................. 11-20
   onmode -wf, -wm: Dynamically change certain configuration parameters 11-20
   onmode -wm: Change LRU tuning status ............................ 11-20
   onmode -Y: Dynamically change SET EXPLAIN .................. 11-22
   onmode -z: Kill a database server session ......................... 11-23
   onmode -Z: Kill a distributed transaction ......................... 11-23

Chapter 12. The ON-Monitor Utility .................................... 12-1
Using ON-Monitor (UNIX) ................................................. 12-1
   Navigating ON-Monitor and Using Help ............................ 12-1
   Executing Shell Commands Within ON-Monitor .................. 12-2
Chapter 13. The onparams Utility ........................................... 13-1
In This Chapter ........................................................................ 13-1
onparams Syntax ................................................................. 13-2
onparams -a -d dbspace: Add a logical-log file ......................... 13-2
onparams -d -l lognum: Drop a logical-log file ......................... 13-3
onparams -p: Change physical-log parameters ......................... 13-3
  Backing Up After You Change the Physical-Log Size or Location .. 13-4
  Changing the Size of the Physical Log and Using Non-Default Page Sizes . 13-4
  Using a Text Editor to Change the Physical-Log Size or Location . 13-4
onparams -b: Add a new buffer pool ...................................... 13-4
Examples of onparams Commands ........................................... 13-6

Chapter 14. The onspaces Utility ............................................. 14-1
In This Chapter ........................................................................ 14-1
onspaces Syntax ...................................................................... 14-2
Administrative API command string ....................................... 14-2
onspaces -a: Add a chunk to a dbspace or blobspace ................. 14-3
onspaces -a: Add a chunk to an sbspace .................................. 14-4
onspaces -c -q: Create a blobspace ....................................... 14-6
onspaces -c -d: Create a dbspace ........................................... 14-8
  Creating a Temporary Dbspace with the -t Option .................. 14-10
  Specifying First and Next Extent Size for the tbspace tbspace ...... 14-11
  Specifying a Non-Default Page Size with the Same Size as the Buffer Pool 14-11
onspaces -c -g: Create an sbspace ........................................... 14-11
  Creating a Temporary Sbspace with the -t Option ................. 14-13
  Creating an Sbspace with the -Df option ...................... 14-13
  Changing the -Df Settings ................................................ 14-16
  Using the onspaces -g Option ........................................... 14-16
onspaces -c -x: Create an extspace ....................................... 14-16
onspaces -c: Change sbspace default specifications ................ 14-17
onspaces -c: Clean up stray smart large objects in sspaces ......... 14-18
onspaces -d: Drop a chunk in a dbspace, blobspace, or sbspace ... 14-19
onspaces -d: Drop a blobspace, dbspace, extspace, or sbspace ... 14-20
onspaces -f: Specify DATASKIP parameter ................................ 14-21
onspaces -m: Start mirroring ................................................ 14-22
  Using a File to Specify Chunk-Location Information with the -f Option . 14-23
onspaces -r: Stop mirroring ................................................... 14-24
onspaces -ren: Rename a dbspace, blobspace, sbspace, or extspace . 14-24
  Renaming a Dbspace, Blobspace, Sbspace, or Extspace When Enterprise Replication Is Active . 14-25
  Performing an Archive After Renaming a Space ..................... 14-25
  Performing an Archive After Renaming a Space ..................... 14-25
onspaces -s: Change status of a mirrored chunk ...................... 14-26

Chapter 15. The onstat Utility ................................................. 15-1
In This Chapter ........................................................................ 15-3
Monitor the Database Server Status ...................................... 15-4
onstat Syntax ......................................................................... 15-4
  Statistics Collected from Source File .................................. 15-7
  Interactive Execution ....................................................... 15-8
  Continuous onstat Execution ............................................ 15-8
onstat: onstat -pu equivalent .............................................. 15-8
onstat : Print output header ............................................... 15-8
  Logs Full Subheader ....................................................... 15-9
onstat -- Print onstat options and functions ......................... 15-9
onstat <option> <infile> : Print shared memory dump file ......... 15-9
onstat -a: Print onstat -cusbtdlp ........................................ 15-10
onstat -b: Print buffer information ...................................... 15-10
onstat -B: Print all buffer information ................................... 15-12
onstat -c: Print ONCONFIG file contents .............................................. 15-13
onstat -C: Print B-tree scanner information ......................................... 15-13
onstat -d: Print chunk information ..................................................... 15-14
    Using onstat -d with Sspaces ................................................... 15-17
    Using onstat -d with Blobspaces .............................................. 15-17
onstat -D: Print page-read and page-write information ......................... 15-18
onstat -f: Print dbspace information affected by dataskip .................. 15-18
onstat -F: Print counts ................................................................. 15-19
onstat -g Monitoring Options .......................................................... 15-20
    onstat -g act: Print active threads ........................................... 15-25
    onstat -g afr: Print allocated memory fragments ........................... 15-26
    onstat -g all: Print output from all onstat -g options .................. 15-26
    onstat -g ath: Print information about all threads ...................... 15-26
    onstat -g buf: Print buffer pool profile information ................... 15-27
    onstat -g cat: Print ER global catalog information ...................... 15-29
    onstat -g cdr config: Print ER settings ...................................... 15-30
    onstat -g con: Print condition and thread information ................. 15-32
    onstat -g ddr: Print ER database log reader status ...................... 15-33
    onstat -g dic: Print table information ....................................... 15-34
    onstat -g dis: Print database server information ........................ 15-35
    onstat -g dll: Print dynamic libraries list .................................. 15-36
    onstat -g dmp: Print raw memory ............................................... 15-37
    onstat -g dri: Print HDR information ......................................... 15-38
    onstat -g dsc: Print distribution cache information ...................... 15-38
    onstat -g dss: Print ER environment data .................................... 15-39
    onstat -g dtc: Print delete table cleaner statistics ...................... 15-40
    onstat -g env: Prints environment variable values ......................... 15-41
    onstat -g ffr: Print free fragments ............................................ 15-43
    onstat -g glo: Print global multithreading information .................. 15-44
    onstat -g grp: Print ER grouper statistics .................................. 15-45
    onstat -g his: Print SQLTRACE information .................................. 15-50
    onstat -g ioa: Print combined onstat -g information ...................... 15-52
    onstat -g job: Print big buffer use summary .................................. 15-53
    onstat -g iof: Print asynchronous I/O statistics ............................ 15-54
    onstat -g iog: Print AIO global information .................................. 15-55
    onstat -g ioc: Print I/O queue information .................................. 15-55
    onstat -g ipl: Print index page logging status information ............. 15-57
    onstat -g iov: Print AIO VP statistics ........................................ 15-57
    onstat -g lap: Print light appends status information ...................... 15-58
    onstat -g lm: Print all locked mutexes ....................................... 15-59
    onstat -g lsc: Print active light scan status ................................ 15-60
    onstat -g mem: Print pool memory statistics .................................. 15-60
    onstat -g mgm: Print MGM resource information ............................... 15-61
    onstat -g nbm: Print a block bit map ......................................... 15-64
    onstat -g nif: Print ER network interface statistics ...................... 15-65
    onstat -g nsc: client_id: Print current shared memory connection information 15-66
    onstat -g snsd: Print poll threads shared-memory data ................... 15-68
    onstat -g nss: Print shared memory connections status .................... 15-69
    onstat -g ntd: Print network statistics ....................................... 15-70
    onstat -g ntm: Print network mail statistics .................................. 15-70
    onstat -g nt: Print network user times ....................................... 15-71
    onstat -g ntu: Print network user statistics .................................. 15-71
    onstat -g opn: Print open partitions ......................................... 15-72
    onstat -g pos: Print file values ............................................... 15-74
    onstat -g pfp: partition number | 0: Print partition profiles ........ 15-75
    onstat -g prc: Print sessions using UDR or SPL routine .................. 15-76
    onstat -g que: Prints ER queue statistics .................................... 15-76
    onstat -g qst: Print vait options for mutex and condition queues ..... 15-77
    onstat -g rbm: Print a block map of shared memory ....................... 15-78
    onstat -g rvc: Print ER receive manager statistics ....................... 15-79
    onstat -g re: Print ready threads ............................................. 15-82

Contents xi
onstat -g rep: Print ER schedule manager events ........................................... 15-82
onstat -g rqm: Print low-level queue statistics ............................................. 15-82
onstat -g rss: Print RS secondary server information .................................. 15-85
onstat -g rwm: Print read and write mutexes ............................................... 15-88
onstat -g sch: Print VP information ............................................................. 15-89
onstat -g sds: Print SDS secondary server information ............................... 15-93
onstat -g seg: Print shared-memory segment statistics ................................ 15-93
onstat -g ses: Print session-related information ........................................... 15-98
onstat -g sle: Print all sleeping threads ...................................................... 15-98
onstat -g smb: Print sbspaces information ................................................... 15-99
onstat -g smx: Print multiplexer group information .................................... 15-101
onstat -g spi: Print spin locks with long spins .............................................. 15-102
onstat -g sql: Print SQL-related session information .................................. 15-103
onstat -g src: Patterns in shared memory ..................................................... 15-104
onstat -g ssc: Print SQL statement occurrences .......................................... 15-106
onstat -g stk tid: Print thread stack ......................................................... 15-106
onstat -g stm: Print SQL statement memory usage ....................................... 15-107
onstat -g stq: Print queue information ....................................................... 15-107
onstat -g stx: Print stack usage per thread ................................................. 15-107
onstat -g sync: Print ER synchronization status ......................................... 15-108
onstat -g tpf tid: Print thread profiles ....................................................... 15-109
onstat -g ufr: Print memory pool fragments ................................................. 15-109
onstat -g vpcache: Print CPU VP memory block cache statistics .................. 15-110
onstat -g wai: Print wait queue thread list ................................................ 15-110
onstat -g wmx: Print all mutexes with waiters .......................................... 15-113
onstat -g wst: Print wait statistics for threads .......................................... 15-113
onstat -G: Print TP/XA transaction information .......................................... 15-114
onstat -h: Print buffer header hash chain information ............................... 15-114
onstat -i: Initiate interactive mode .............................................................. 15-115
onstat -j: Provide onload status information ............................................... 15-115
onstat -k: Print active lock information ..................................................... 15-115
onstat -l: Print physical and logical log information ................................... 15-115
onstat -m: Print recent system message log information ............................ 15-116
onstat -o: Output shared memory contents ................................................. 15-116
onstat -O: Print optical subsystem information ......................................... 15-116
onstat -p: Print profile counts ................................................................. 15-117
onstat -P: Print partition information .......................................................... 15-117
onstat -r: Repeatedly print selected statistics ............................................ 15-118
onstat -R: Print LRU, FLRU, and MLRU queue information ......................... 15-118
onstat -s: Print latch information ............................................................... 15-118
onstat -t and -T: Print tblspace information ............................................. 15-118
onstat -u: Print user activity profile ........................................................... 15-118
onstat -x: Print database server transaction information ........................... 15-119
onstat -X: Print thread information ............................................................. 15-120
onstat -z: Clear statistics ....................................... .
Contents

Appendix E. Error Messages .............................................. E-1

Appendix F. Limits in IBM Informix Dynamic Server ................. F-1
Limitations on UNIX Operating Systems ................................ F-1
  System-Level Parameter Limits (UNIX) .................................. F-1
  Table-Level Parameter Limits (UNIX) .................................... F-1
  Access Capabilities (UNIX) ................................................ F-1
  IBM Informix Dynamic Server System Defaults (UNIX) ............... F-2
  ON-Monitor Statistics (UNIX) ............................................ F-2
Limitations on Windows Operating Systems .............................. F-3
  System-Level Parameter Limits (Windows) ............................... F-3
  Table-Level Parameter Limits (Windows) ............................... F-3
  Access Capabilities (Windows) .......................................... F-3
  IBM Informix Dynamic Server System Defaults (Windows) .......... F-4

Appendix G. Accessibility ................................................. G-1
Accessibility features for IBM Informix Dynamic Server ............ G-1
  Accessibility Features .................................................. G-1
  Keyboard Navigation .................................................... G-1
  Related Accessibility Information ...................................... G-1
  IBM and Accessibility .................................................. G-1
Dotted Decimal Syntax Diagrams ......................................... G-1

Notices .......................................................................... H-1
Trademarks ..................................................................... H-3

Index ............................................................................. X-1
Introduction

In This Introduction ................................................................. xv
About This Publication .............................................................. xv
Types of Users .............................................................................. xv
  Software Dependencies ........................................................... xvi
  Assumptions About Your Locale .............................................. xvi
  Demonstration Database ........................................................ xvi
New Features ................................................................................ xvi
Documentation Conventions ..................................................... xix
  Typographical Conventions ..................................................... xix
  Feature, Product, and Platform Markup ................................. xx
  Example Code Conventions ..................................................... xx
Additional Documentation ......................................................... xxi
Compliance with Industry Standards ......................................... xxi
Syntax Diagrams ........................................................................... xxi
  How to Read a Command-Line Syntax Diagram ..................... xxii
  Keywords and Punctuation ..................................................... xxiii
  Identifiers and Names ........................................................... xxiii
IBM Welcomes Your Comments ................................................ xxiv

In This Introduction

This introduction provides an overview of the information in this publication and describes the conventions it uses.

About This Publication

This publication provides reference material for IBM Informix Dynamic Server. It contains comprehensive descriptions of configuration parameters, the system-monitoring interface (SMI) tables in the sysmaster database, the syntax of database server utilities such as onmode and onstat, logical-log records, disk structures, event alarms, and unnumbered error messages. This publication has two companion volumes, the IBM Informix Administrator’s Guide and the IBM Informix Performance Guide.

This section discusses the intended audience for this publication and the associated software products that you must have to use the administrative utilities.

Types of Users

This publication is written for the following users:
• Database administrators
• System administrators
• Performance engineers

This publication is written with the assumption that you have the following background:
• A working knowledge of your computer, your operating system, and the utilities that your operating system provides
• Some experience working with relational databases or exposure to database concepts
Some experience with database server administration, operating-system administration, or network administration

If you have limited experience with relational databases, SQL, or your operating system, refer to the IBM Informix Getting Started Guide for your database server for a list of supplementary titles.

Software Dependencies
This publication is written with the assumption that you are using IBM Informix Dynamic Server or IBM Informix Dynamic Server with J/Foundation, Version 11.1, as your database server.

Assumptions About Your Locale
IBM Informix products can support many languages, cultures, and code sets. All the information related to character set, collation, and representation of numeric data, currency, date, and time is brought together in a single environment, called a Global Language Support (GLS) locale.

The examples in this publication are written with the assumption that you are using the default locale, en_us.8859-1. This locale supports U.S. English format conventions for date, time, and currency. In addition, this locale supports the ISO 8859-1 code set, which includes the ASCII code set plus many 8-bit characters such as é, è, and ň.

If you plan to use nondefault characters in your data or your SQL identifiers, or if you want to conform to the nondefault collation rules of character data, you need to specify the appropriate nondefault locale.

For instructions on how to specify a nondefault locale, additional syntax, and other considerations related to GLS locales, see the IBM Informix GLS User’s Guide.

Demonstration Database
The DB–Access utility, which is provided with your Informix® database server products, includes one or more of the following demonstration databases:

- The stores_demo database illustrates a relational schema with information about a fictitious wholesale sporting-goods distributor. Many examples in IBM Informix publications are based on the stores_demo database.
- The superstores_demo database illustrates an object-relational schema. The superstores_demo database contains examples of extended data types, type and table inheritance, and user-defined routines.

For information about how to create and populate the demonstration databases, see the IBM Informix DB–Access User’s Guide. For descriptions of the databases and their contents, see the IBM Informix Guide to SQL: Reference.

The scripts that you use to install the demonstration databases reside in the $INFORMIXDIR/bin directory on UNIX® and in the %INFORMIXDIR%\bin directory on Windows.

New Features
For a comprehensive list of new features for this release, see the IBM Informix Getting Started Guide. This topic lists new features relevant to this publication.
<table>
<thead>
<tr>
<th>New Features</th>
<th>Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automatic checkpoint and bufferpool flushing. See</td>
<td>A patented checkpoint algorithm that virtually eliminates transaction blocking during checkpoints. This feature allows the server to automatically control the frequency of checkpoints to avoid transaction blocking, and also automatically adjusts LRU flushing. In addition, the server attempts to automatically adjust the number of AIO VPs and page-clearer threads for optimal flushing performance.</td>
</tr>
<tr>
<td>• “AUTO_AIOVPS” on page 1-15</td>
<td></td>
</tr>
<tr>
<td>• “AUTO_CKPTS” on page 1-16</td>
<td></td>
</tr>
<tr>
<td>• “AUTO_LRU_TUNING” on page 1-16</td>
<td></td>
</tr>
<tr>
<td>“ADMIN_MODE_USERS” on page 1-13</td>
<td>A configuration parameter that specifies which users (in addition to user informix and DBSA group users) can access the database server in administration mode.</td>
</tr>
<tr>
<td>“ADMIN_USER_MODE_WITH_DBSA” on page 1-13</td>
<td>A configuration parameter that specifies whether user informix and the DBSA group users can connect to the database server while it is in administration mode.</td>
</tr>
<tr>
<td>“AUTO_REPREPARE” on page 1-16</td>
<td>A configuration parameter that automatically re-optimizes SPL routines and re-prepares prepared objects after the schema of a table referenced by the SPL routine or by the prepared object has been changed.</td>
</tr>
<tr>
<td>“DIRECT_IO (UNIX)” on page 1-32</td>
<td>A configuration parameter that controls the use of direct I/O for cooked files used for dbspace chunks.</td>
</tr>
<tr>
<td>“DRDA_COMMBUFFSIZE” on page 1-33</td>
<td>A configuration parameter that specifies the size of the DRDA® communications buffer.</td>
</tr>
<tr>
<td>ENCRYPT_HDR</td>
<td>A configuration parameter that enables you to encrypt data traffic between the servers in an HDR pair.</td>
</tr>
<tr>
<td>EXPLAIN_STAT</td>
<td>A configuration parameter that enables you to display query statistics information in your explain output file.</td>
</tr>
<tr>
<td>IFX_FOLDVIEW</td>
<td>A configuration parameter that enables view folding, which can significantly improve the performance of the query.</td>
</tr>
<tr>
<td>LOG_INDEX_BUILDS</td>
<td>A configuration parameter that enables index page logging, which writes newly created index files to the logical log for the purpose of synchronizing index creation between servers in high-availability environments.</td>
</tr>
<tr>
<td>Enhancements to the onmode utility. See onmode -wf,-wm: Change LRU tuning status</td>
<td>The onmode utility has been enhanced to support shared disk (SD) secondary servers.</td>
</tr>
<tr>
<td>Enhancements to Chapter 15, “The onstat Utility”</td>
<td>The onstat utility has been enhanced to support shared disk (SD) and remote standalone (RS) secondary servers.</td>
</tr>
<tr>
<td></td>
<td>In addition, the following options have been included:</td>
</tr>
<tr>
<td>• “onstat -o: Output shared memory contents” on page 15-125</td>
<td></td>
</tr>
<tr>
<td>• “onstat -r: Repeatedly print selected statistics” on page 15-131</td>
<td></td>
</tr>
</tbody>
</table>
**New Features**

Enhancements to **onstat -g Monitoring Options**

The **onstat -g** utility option has been enhanced with the following:

- `onstat -g <option> <infile>:` Print shared memory dump file
- `onstat -g all:` Print output from all onstat -g options
- `onstat -g buf:` Print buffer pool profile information
- `onstat -g cdr config:` Print ER settings
- `onstat -g dmp:` Print raw memory
- `onstat -g lap:` Print light appends status information
- `onstat -g lsc:` Print active light scan status
- `onstat -g nss:` Print shared memory network connections status
- `onstat -g opn:` Print open partitions
- `onstat -g qst:` Print wait options for mutex and condition queues
- `onstat -g rbm:` Print a block map of shared memory
- `onstat -g spi:` Print spin locks with long spins
- `onstat -g src:` Patterns in shared memory
- `onstat -g stq:` Print queue information
- `onstat -g ufr:` Print memory pool fragments
- `onstat -g wai:` Print wait queue thread list
- `onstat -g wst:` Print wait statistics for threads

**RTO_SERVER_RESTART**

A configuration parameter that enables you to set the amount of time, in seconds, that Dynamic Server has to recover from a problem after you restart Dynamic Server and bring the server into online or quiescent mode.

**"SDS_ENABLE" on page 1-84**

**"SDS_PAGING" on page 1-84**

**"SDS_TEMPDBS" on page 1-84**

**"SDS_TIMEOUT" on page 1-85**

**"SHMVIRT_ALLOCSEG" on page 1-88**

A configuration parameter that specifies a threshold at which Dynamic Server should allocate server memory and the alarm level activated if the server cannot allocate the new memory segment.
New Features | Overview
---|---
“SQLTRACE” on page 1-91 | A configuration parameter that controls the startup environment of the SQL Trace facility.

“The sysadmin Database” on page 3-1 | The sysadmin database contains the six tables which contain and organize tasks information for the Scheduler, an administrative tool that enables the database server to execute database functions and procedures at predefined times or as determined internally by the server.

Enhancements to “The sysmaster Database” on page 2-2 | The sysmaster database has been enhanced to support new high-availability configuration options. A high-availability configuration consists of a primary server and one or more secondary servers. Multiple types of secondary servers can coexist in a high-availability configuration. A secondary server can be an SD secondary server, an RS secondary server, or an HDR secondary server.

USELASTCOMMITTED | A configuration parameter that can enable improved performance and concurrency in Committed Read and Dirty Read isolation levels when Read operations encounter tables with locks from other open transactions.

“VP_MEMORY_CACHE_KB” on page 1-103 | A configuration parameter that enables the database server to access the private memory blocks of your CPU VP.

Documentation Conventions

This section describes the following conventions, which are used in the product documentation for IBM Informix Dynamic Server:
- Typographical conventions
- Feature, product, and platform conventions
- Syntax diagrams
- Command-line conventions
- Example code conventions

Typographical Conventions

This publication uses the following conventions to introduce new terms, illustrate screen displays, describe command syntax, and so forth.

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEYWORD</td>
<td>Keywords of SQL, SPL, and some other programming languages appear in uppercase letters in a serif font.</td>
</tr>
<tr>
<td><em>italics</em></td>
<td>Within text, new terms and emphasized words appear in italics. Within syntax and code examples, variable values that you are to specify appear in italics.</td>
</tr>
<tr>
<td><em>boldface</em></td>
<td>Names of program entities (such as classes, events, and tables), environment variables, file names, path names, and interface elements (such as icons, menu items, and buttons) appear in boldface.</td>
</tr>
<tr>
<td><em>monospace</em></td>
<td>Information that the product displays and information that you enter appear in a monospace typeface.</td>
</tr>
<tr>
<td>Convention</td>
<td>Meaning</td>
</tr>
<tr>
<td>------------</td>
<td>---------</td>
</tr>
<tr>
<td>KEYSSTROKE</td>
<td>Keys that you are to press appear in uppercase letters in a sans serif font.</td>
</tr>
<tr>
<td>&gt;</td>
<td>This symbol indicates a menu item. For example, “Choose Tools &gt; Options” means choose the Options item from the Tools menu.</td>
</tr>
</tbody>
</table>

**Feature, Product, and Platform Markup**

Feature, product, and platform markup identifies paragraphs that contain feature-specific, product-specific, or platform-specific information. Some examples of this markup follow:

### Dynamic Server

Identifies information that is specific to IBM Informix Dynamic Server

### End of Dynamic Server

### Windows Only

Identifies information that is specific to the Windows operating system

### End of Windows Only

This markup can apply to one or more paragraphs within a section. When an entire section applies to a particular product or platform, this is noted as part of the heading text, for example:

**Table Sorting (Windows)**

**Example Code Conventions**

Examples of SQL code occur throughout this publication. Except as noted, the code is not specific to any single IBM Informix application development tool.

If only SQL statements are listed in the example, they are not delimited by semicolons. For instance, you might see the code in the following example:

```sql
CONNECT TO stores_demo
...
DELETE FROM customer
   WHERE customer_num = 121
...
COMMIT WORK
DISCONNECT CURRENT
```

To use this SQL code for a specific product, you must apply the syntax rules for that product. For example, if you are using DB–Access, you must delimit multiple statements with semicolons. If you are using an SQL API, you must use EXEC SQL at the start of each statement and a semicolon (or other appropriate delimiter) at the end of the statement.

**Tip:** Ellipsis points in a code example indicate that more code would be added in a full application, but it is not necessary to show it to describe the concept being discussed.
For detailed directions on using SQL statements for a particular application development tool or SQL API, see the documentation for your product.

Additional Documentation

You can view, search, and print all of the product documentation from the IBM Informix Dynamic Server information center on the Web at:

```
publib.boulder.ibm.com/infocenter/idshelp/v111/index.jsp
```

For additional documentation about IBM Informix Dynamic Server and related products, including release notes, machine notes, and documentation notes, go to the online product library page at:

```
```
Alternatively, you can access or install the product documentation from the Quick Start CD that is shipped with the product.

Compliance with Industry Standards

The American National Standards Institute (ANSI) and the International Organization of Standardization (ISO) have jointly established a set of industry standards for the Structured Query Language (SQL). IBM® Informix SQL-based products are fully compliant with SQL-92 Entry Level (published as ANSI X3.135-1992), which is identical to ISO 9075:1992. In addition, many features of IBM Informix database servers comply with the SQL-92 Intermediate and Full Level and X/Open SQL Common Applications Environment (CAE) standards.

Syntax Diagrams

This guide uses syntax diagrams built with the following components to describe the syntax for statements and all commands other than system-level commands.

<table>
<thead>
<tr>
<th>Component represented in PDF</th>
<th>Component represented in HTML</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>↘</td>
<td>&gt;&gt;--------------------------&lt;</td>
<td></td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>&gt;&gt;&gt;--------------------------</td>
<td>&gt;&gt;--------------------------&lt;</td>
<td></td>
</tr>
<tr>
<td>&gt;-----------------------------</td>
<td>&gt;-----------------------------</td>
<td>Statement continues from previous line.</td>
</tr>
<tr>
<td>&gt;-----------------------------</td>
<td>&gt;-----------------------------</td>
<td>Statement ends.</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>---SELECT---------------------</td>
<td>-------SELECT-------&lt;</td>
<td>Required item.</td>
</tr>
<tr>
<td>LOCAL</td>
<td>'-----LOCAL------'</td>
<td>Optional item.</td>
</tr>
<tr>
<td>ALL</td>
<td>+++--ALL-------++</td>
<td>Required item with choice.</td>
</tr>
<tr>
<td>DISTINCT</td>
<td>++DISTINCT------++</td>
<td>One and only one item must be present.</td>
</tr>
</tbody>
</table>
| UNIQUE                       | '---UNIQUE------'            | -------
Table 1. Syntax Diagram Components (continued)

<table>
<thead>
<tr>
<th>Component represented in PDF</th>
<th>Component represented in HTML</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOR UPDATE ---- FOR READ ONLY ----</td>
<td>---+------------------+---</td>
<td>Optional items with choice are shown below the main line, one of which you might specify.</td>
</tr>
</tbody>
</table>
| NEXT ---- PRIOR ---- PREVIOUS ---- | .---NEXT--------.  
| index_name ---- table_name ---- | .-------,-----------.  
| Table Reference ---- | Table Reference | Reference to a syntax segment. |
| view ---- table ---- synonym ---- | Table Reference  
| | | Syntax segment. |

How to Read a Command-Line Syntax Diagram

The following command-line syntax diagram uses some of the elements listed in the table in Syntax Diagrams.

Creating a No-Conversion Job

| onpladm create job---job | -p--project---n--d--device---D--database | t--table | Setting the Run Mode | (1) |

Notes:

1. See page Z-1

The second line in this diagram has a segment named “Setting the Run Mode,” which according to the diagram footnote, is on page Z-1. If this was an actual
cross-reference, you would find this segment in on the first page of Appendix Z. Instead, this segment is shown in the following segment diagram. Notice that the diagram uses segment start and end components.

Setting the Run Mode:

To see how to construct a command correctly, start at the top left of the main diagram. Follow the diagram to the right, including the elements that you want. The elements in this diagram are case sensitive because they illustrate utility syntax. Other types of syntax, such as SQL, are not case sensitive.

The Creating a No-Conversion Job diagram illustrates the following steps:
1. Type `onpladm create job` and then the name of the job.
2. Optionally, type `-p` and then the name of the project.
3. Type the following required elements:
   - `-n`
   - `-d` and the name of the device
   - `-D` and the name of the database
   - `-t` and the name of the table
4. Optionally, you can choose one or more of the following elements and repeat them an arbitrary number of times:
   - `-S` and the server name
   - `-T` and the target server name
   - The run mode. To set the run mode, follow the Setting the Run Mode segment diagram to type `-f`, optionally type `d, p, or a`, and then optionally type `I` or `u`.
5. Follow the diagram to the terminator.

Keywords and Punctuation

Keywords are words reserved for statements and all commands except system-level commands. When a keyword appears in a syntax diagram, it is shown in uppercase letters. When you use a keyword in a command, you can write it in uppercase or lowercase letters, but you must spell the keyword exactly as it appears in the syntax diagram.

You must also use any punctuation in your statements and commands exactly as shown in the syntax diagrams.

Identifiers and Names

Variables serve as placeholders for identifiers and names in the syntax diagrams and examples. You can replace a variable with an arbitrary name, identifier, or literal, depending on the context. Variables are also used to represent complex syntax elements that are expanded in additional syntax diagrams. When a variable appears in a syntax diagram, an example, or text, it is shown in lowercase italic.
The following syntax diagram uses variables to illustrate the general form of a simple SELECT statement.

```
SELECT column_name FROM table_name
```

When you write a SELECT statement of this form, you replace the variables `column_name` and `table_name` with the name of a specific column and table.

---

**IBM Welcomes Your Comments**

We want to know about any corrections or clarifications that you would find useful in our publications, which will help us improve future versions. Include the following information:

- The name and version of the publication that you are using
- Section and page number
- Your suggestions about the publication

Send your comments to us at the following e-mail address:

[docinf@us.ibm.com](mailto:docinf@us.ibm.com)

This e-mail address is reserved for reporting errors and omissions in our documentation. For immediate help with a technical problem, contact IBM Technical Support. For instructions, see the IBM Informix Technical Support website at [http://www.ibm.com/planetwide/](http://www.ibm.com/planetwide/)

We appreciate your suggestions.
Part 1. Configuring and Monitoring Dynamic Server
Chapter 1. Configuration Parameters

In This Chapter ..................................................... 1-4
ONCONFIG File Conventions ........................................ 1-4
   Format of ONCONFIG File ...................................... 1-4
   ONCONFIG File Templates ...................................... 1-4
   Printing the onconfig.std File ................................. 1-5
Specifying Hidden Configuration Parameters .................... 1-5
Displaying ONCONFIG Settings .................................... 1-5
Summary of Configuration Parameters ............................. 1-6
   Parameter Attributes ........................................... 1-11
   Using a Utility to Change a Parameter Value ............... 1-11
   Environment Variables ....................................... 1-12
Archecker Configuration Parameters ............................... 1-12
ADMIN_MODE_USERS ............................................. 1-13
ADMIN_USER_MODE_WITH_DBSA ................................... 1-13
ADTERR, ADTMODE, ADTPATH, and ADTSIZE (UNIX) ........... 1-14
ALARMPROGRAM .................................................. 1-14
ALLOW_NEWLINE .................................................. 1-14
ALRM_ALL_EVENTS .............................................. 1-15
AUTO_AIOVPS .................................................... 1-15
AUTO_CKPTS ...................................................... 1-16
AUTO_LRU_TUNING ............................................... 1-16
AUTO_REPREPAIR ................................................ 1-16
BLOCKTIMEOUT .................................................. 1-17
BTSCANNER ....................................................... 1-17
BUFFERPOOL ...................................................... 1-18
   The Irs Field .................................................. 1-20
   The buffers Field ............................................. 1-20
      Buffers and Read-Ahead .................................... 1-21
      Buffers and Smart Large Objects ......................... 1-21
   The lru_min_dirty Field ..................................... 1-21
   The lru_max_dirty Field ..................................... 1-22
   The size Field ............................................... 1-22
      System Page Size .......................................... 1-22
CKPTINTVL ....................................................... 1-22
CLEANERS ........................................................ 1-23
CONSOLE ........................................................ 1-24
DATASKIP ........................................................ 1-24
DBCREATE_PERMISSION ......................................... 1-25
DB_LIBRARY_PATH ............................................... 1-25
DBSERVERALIASES .............................................. 1-26
DBSERVERNAME ................................................. 1-27
DBSPACETEMP .................................................... 1-27
   Using Hash Join Overflow and DBSPACETEMP .............. 1-29
DD_HASHMAX ...................................................... 1-30
DD_HASHSIZE ..................................................... 1-30
DEADLOCK_TIMEOUT .............................................. 1-30
DEF_TABLE_LOCKMODE .......................................... 1-31
DIRECT_IO (UNIX) ............................................... 1-32
DIRECTIVES ..................................................... 1-32
DISABLE_B162428_XA_FIX ....................................... 1-33
DRDA_COMMBUFFSIZE ........................................... 1-33
DRAUTO .......................................................... 1-34
DRDUXAUTO ...................................................... 1-35
DRINTERVAL ..................................................... 1-35
DRLOSTFOUND .................................................... 1-35

© Copyright IBM Corp. 1996, 2007 1-1
In This Chapter

This chapter describes the ONCONFIG file conventions, lists the configuration parameters in the ONCONFIG file, and provides a short discussion of each parameter.

ONCONFIG File Conventions

The ONCONFIG environment variable specifies the file that contains the configuration parameters. This file is also called the ONCONFIG file. The database server uses the ONCONFIG file during initialization.

Format of ONCONFIG File

In the ONCONFIG file, each parameter is on a separate line. The file can also contain blank lines and comment lines that start with a # symbol. The following line shows the syntax for a parameter line:

PARAMETER_NAME parameter_value #comment

Parameters and their values in the ONCONFIG file are case sensitive. The parameter names are always uppercase. If the value entry is described with uppercase letters, you must use uppercase (for example, the CPU value of the NETTYPE parameter). You must put white space (tabs, spaces, or both) between the parameter name, parameter value, and optional comment. Do not use any tabs or spaces within a parameter value.

Restriction: The maximum line limit of the ONCONFIG file is 512 bytes. Lines that exceed this limit are truncated and might cause configuration problems.

ONCONFIG File Templates

The database server provides a template for a configuration file that contains initial values for many of the ONCONFIG parameters.

IBM Informix Dynamic Server provides onconfig.std as a template configuration file that you can copy and tailor to your specific configuration.

If you omit a parameter value in your copy of the configuration file, the database server either uses default values in onconfig.std or calculates values based on other parameter values. For information on the order of files in which the database server looks for configuration values during initialization, refer to the chapter on initializing the database server in the IBM Informix Administrator’s Guide.
**Warning:** Do not modify or delete onconfig.std, which is a template and not a functional configuration.

The following table lists the locations of the ONCONFIG and onconfig.std files.

<table>
<thead>
<tr>
<th>Operating System</th>
<th>ONCONFIG File</th>
<th>Template File</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIX</td>
<td>$INFORMIXDIR/etc/$ONCONFIG</td>
<td>$INFORMIXDIR/etc/onconfig.std</td>
</tr>
<tr>
<td>Windows</td>
<td>%INFORMIXDIR%\etc%ONCONFIG%</td>
<td>%INFORMIXDIR%\etc\onconfig.std</td>
</tr>
</tbody>
</table>

To prepare the ONCONFIG file:
1. Copy the onconfig.std template file.
2. Modify the copy of the template file.
3. Set the ONCONFIG environment variable to the name of the copy of the pertinent template file.
   - If you do not set ONCONFIG, the default filename is onconfig.

For more details on why you might want to modify the default configuration parameters, refer to the chapter on configuring the database server in the *IBM Informix Administrator's Guide*.

**Printing the onconfig.std File**

**Important:** Print out a copy of the onconfig.std file to see the latest default values for the configuration parameters and recommended settings.

**Specifying Hidden Configuration Parameters**

A few of the configuration parameters, such as DYNAMIC_LOGS, are omitted from the onconfig.std file. It is recommended that you use the default values for these hidden parameters. If you want to change the value for a hidden parameter, add it to your ONCONFIG file.

**Displaying ONCONFIG Settings**

When the database server restarts, it reads the ONCONFIG file. To view the ONCONFIG settings, use one of the following tools:

- IBM Informix Server Administrator (ISA)
- oncheck -pr

   The information under PAGE_CONFIG lists the configuration parameter settings at restart. For more information, see "oncheck -pr and pR: Display reserved-page information" on page 7-18

- .infos.dbservername

   If you set the ONCONFIG environment variable to the name of a different ONCONFIG file while the database server is online, the .infos.dbservername file contains the current settings. For more information, see "The ONCONFIG File" on page A-7

For more information about the ONCONFIG environment variable, see the *IBM Informix Guide to SQL: Reference*. 

Chapter 1. Configuration Parameters 1-5
Summary of Configuration Parameters

This section provides the following information:

- A list of each configuration parameter with database server compatibility
- A description of the attributes listed for each configuration parameter

The configuration parameters and database server compatibility are as follows. For information on the discontinued configuration parameters, see Appendix D. If the configuration parameter has a related environment variable, it is listed in the following table.

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Related Environment Variable</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC_DEBUG</td>
<td>AC_CONFIG</td>
<td>page 1-12</td>
</tr>
<tr>
<td>AC_INDEX</td>
<td>AC_CONFIG</td>
<td>page 1-12</td>
</tr>
<tr>
<td>AC_LTAPEBLOCK</td>
<td>AC_CONFIG</td>
<td>page 1-12</td>
</tr>
<tr>
<td>AC_LTAPEDEV</td>
<td>AC_CONFIG</td>
<td>page 1-12</td>
</tr>
<tr>
<td>AC_MSPATH</td>
<td>AC_CONFIG</td>
<td>page 1-12</td>
</tr>
<tr>
<td>AC_SCHEMA</td>
<td>AC_CONFIG</td>
<td>page 1-12</td>
</tr>
<tr>
<td>AC_STORAGE</td>
<td>AC_CONFIG</td>
<td>page 1-12</td>
</tr>
<tr>
<td>AC_TAPEBLOCK</td>
<td>AC_CONFIG</td>
<td>page 1-12</td>
</tr>
<tr>
<td>AC_TAPEDEV</td>
<td>AC_CONFIG</td>
<td>page 1-12</td>
</tr>
<tr>
<td>AC_TIMEOUT</td>
<td>AC_CONFIG</td>
<td>page 1-13</td>
</tr>
<tr>
<td>AC_VERBOSE</td>
<td>AC_CONFIG</td>
<td>page 1-13</td>
</tr>
<tr>
<td>ADMIN_MODE_USERS</td>
<td></td>
<td>page 1-13</td>
</tr>
<tr>
<td>ADMIN_USER_MODE_WITH_DBSA</td>
<td></td>
<td>page 1-13</td>
</tr>
<tr>
<td>ADTERR</td>
<td></td>
<td>page 1-14</td>
</tr>
<tr>
<td>ADTQUEUE</td>
<td></td>
<td>page 1-14</td>
</tr>
<tr>
<td>ADTQUEUE</td>
<td></td>
<td>page 1-14</td>
</tr>
<tr>
<td>ADTSIZE</td>
<td></td>
<td>page 1-14</td>
</tr>
<tr>
<td>AFCRASH</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>AFF_NPROCS</td>
<td></td>
<td>page D-1</td>
</tr>
<tr>
<td>AFF_SPROC</td>
<td></td>
<td>page D-2</td>
</tr>
<tr>
<td>ALARMPROGRAM</td>
<td></td>
<td>page 1-14</td>
</tr>
<tr>
<td>ALLOW_NEWLINE</td>
<td></td>
<td>page 1-14</td>
</tr>
<tr>
<td>ALRM_ALL_EVENTS</td>
<td></td>
<td>page 1-15</td>
</tr>
<tr>
<td>AUTO_AIOVPS</td>
<td></td>
<td>page 1-15</td>
</tr>
<tr>
<td>AUTO_CKPTS</td>
<td></td>
<td>page 1-16</td>
</tr>
<tr>
<td>AUTO_LRU_TUNING</td>
<td></td>
<td>page 1-16</td>
</tr>
<tr>
<td>AUTO_REPREPARE</td>
<td></td>
<td>page 1-16</td>
</tr>
<tr>
<td>BAR_ACT_LOG</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BAR_BSA_LIB_PATH</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BAR_DEBUG</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BAR_DEBUG_LOG</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BAR_HISTORY</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>Configuration Parameter</td>
<td>Related Environment Variable</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>-----------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>BAR_MAX_BACKUP</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BAR_NB_XPORT_COUNT</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BAR_PERFORMANCE</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BAR_PROGRESS_FREQ</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BAR_RETRY</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BAR_XFER_BUF_SIZE</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>BLOCKTIMEOUT</td>
<td></td>
<td>page 1-17</td>
</tr>
<tr>
<td>BTSCANNER</td>
<td></td>
<td>page 1-17</td>
</tr>
<tr>
<td>BUFFERPOOL</td>
<td></td>
<td>page 1-18</td>
</tr>
<tr>
<td>CDR_DBSPACE</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_DSLOCKWAIT</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_ENV</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_EVALTHREADS</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_MAX_DYNAMIC_LOGS</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_NIFCOMPRESS</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_QDATA_SBSPACE</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_QHDR_DBSPACE</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_QUEUEMEM</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_SERIAL</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CDR_SUPPRESS_ATSRISWARN</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>CKPTINTVL</td>
<td></td>
<td>page 1-22</td>
</tr>
<tr>
<td>CLEANERS</td>
<td></td>
<td>page 1-23</td>
</tr>
<tr>
<td>CONSOLE</td>
<td></td>
<td>page 1-24</td>
</tr>
<tr>
<td>DATASKIP</td>
<td></td>
<td>page 1-24</td>
</tr>
<tr>
<td>DB_LIBRARY_PATH</td>
<td></td>
<td>page 1-25</td>
</tr>
<tr>
<td>DBSERVERALIASES</td>
<td></td>
<td>page 1-26</td>
</tr>
<tr>
<td>DBSERVERNAME</td>
<td>INFORMIXSERVER</td>
<td>page 1-27</td>
</tr>
<tr>
<td>DBSPACETEMP</td>
<td>DBSPACETEMP</td>
<td>page 1-27</td>
</tr>
<tr>
<td>DD_HASHMAX</td>
<td></td>
<td>page 1-30</td>
</tr>
<tr>
<td>DD_HASHSIZE</td>
<td></td>
<td>page 1-30</td>
</tr>
<tr>
<td>DEADLOCK_TIMEOUT</td>
<td></td>
<td>page 1-30</td>
</tr>
<tr>
<td>DEF_TABLE_LOCKMODE</td>
<td>IFX_DEF_TABLE_LOCKMODE</td>
<td>page 1-31</td>
</tr>
<tr>
<td>DIRECT_IO</td>
<td></td>
<td>page 1-32</td>
</tr>
<tr>
<td>DIRECTIVES</td>
<td>IFX_DIRECTIVES</td>
<td>page 1-32</td>
</tr>
<tr>
<td>DISABLE_B162428_XA_FIX</td>
<td>IFX_XASTDCOMPLIANCE_XAEND</td>
<td>page 1-33</td>
</tr>
<tr>
<td>DRAUTO</td>
<td></td>
<td>page 1-34</td>
</tr>
<tr>
<td>DRDA_COMMBUFFSIZE</td>
<td></td>
<td>page 1-33</td>
</tr>
<tr>
<td>DRIDXAUTO</td>
<td></td>
<td>page 1-35</td>
</tr>
<tr>
<td>DRINTERVAL</td>
<td></td>
<td>page 1-35</td>
</tr>
<tr>
<td>DRLOSTFOUND</td>
<td></td>
<td>page 1-35</td>
</tr>
<tr>
<td>Configuration Parameter</td>
<td>Related Environment Variable</td>
<td>Reference</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>DRTIMEOUT</td>
<td></td>
<td>page 1-36</td>
</tr>
<tr>
<td>DS_HASHSIZE</td>
<td></td>
<td>page 1-36</td>
</tr>
<tr>
<td>DS_MAX QUERIES</td>
<td></td>
<td>page 1-37</td>
</tr>
<tr>
<td>DS_MAX_SCANS</td>
<td></td>
<td>page 1-37</td>
</tr>
<tr>
<td>DS_NONPDQ_QUERY_MEM</td>
<td></td>
<td>page 1-38</td>
</tr>
<tr>
<td>DS_POOLSIZE</td>
<td></td>
<td>page 1-39</td>
</tr>
<tr>
<td>DS_TOTAL_MEMORY</td>
<td></td>
<td>page 1-39</td>
</tr>
<tr>
<td>DUMPCNT</td>
<td></td>
<td>page 1-40</td>
</tr>
<tr>
<td>DUMPFILE</td>
<td></td>
<td>page 1-41</td>
</tr>
<tr>
<td>DUMPDIR</td>
<td></td>
<td>page 1-41</td>
</tr>
<tr>
<td>DUMPFILE</td>
<td></td>
<td>page 1-42</td>
</tr>
<tr>
<td>DUMPFILE</td>
<td></td>
<td>page 1-42</td>
</tr>
<tr>
<td>DS TOTAL MEMORY</td>
<td></td>
<td>page 1-43</td>
</tr>
<tr>
<td>ENCRYPT_CDR</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>ENCRYPT_CIPHERS</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>ENCRYPTHdr</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>ENCRYPT_HDR</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>ENCRYPT_MAC</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>ENCRYPT_MACFILE</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>ENCRYPT_MACFILE</td>
<td></td>
<td>page 1-49</td>
</tr>
<tr>
<td>ENCRYPT_SMX</td>
<td></td>
<td>page 1-47</td>
</tr>
<tr>
<td>ENCRYPT_SWITCH</td>
<td></td>
<td>page 1-48</td>
</tr>
<tr>
<td>EXPLAIN_STAT</td>
<td></td>
<td>page 1-50</td>
</tr>
<tr>
<td>EXT_DIRECTIVES</td>
<td>IFX_EXTDIRECTIVES</td>
<td>page 1-50</td>
</tr>
<tr>
<td>EXTSHMADD</td>
<td></td>
<td>page 1-51</td>
</tr>
<tr>
<td>FASTPOLL</td>
<td></td>
<td>page 1-51</td>
</tr>
<tr>
<td>FILLFACTOR</td>
<td></td>
<td>page 1-51</td>
</tr>
<tr>
<td>HETERO_COMMIT</td>
<td></td>
<td>page 1-52</td>
</tr>
<tr>
<td>IFX_EXTEND_ROLE</td>
<td></td>
<td>page 1-52</td>
</tr>
<tr>
<td>IMCLOG</td>
<td></td>
<td>page 1-52</td>
</tr>
<tr>
<td>IMCTRANSPORTS</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>IMCWORKERDELAY</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>IMCWORKERDELAY</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>IMCWORKERTHREADS</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>ISM_DATA_POOL</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>Configuration Parameter</td>
<td>Related Environment Variable</td>
<td>Reference</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>ISM_LOG_POOL</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JDKVERSION</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JVMTHREAD</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JVPCLASSPATH</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JVPDEBUG</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JVPHOME</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JVPJAVAHOME</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JVPJAVA LIB</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JVPJAVA VM</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JVPLOGFILE</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>JVPPROFILE</td>
<td></td>
<td>page 1-53</td>
</tr>
<tr>
<td>LISTEN_TIMEOUT</td>
<td></td>
<td>page 1-54</td>
</tr>
<tr>
<td>LOCKS</td>
<td></td>
<td>page 1-54</td>
</tr>
<tr>
<td>LOGBUFF</td>
<td></td>
<td>page 1-55</td>
</tr>
<tr>
<td>LOGFILES</td>
<td></td>
<td>page 1-56</td>
</tr>
<tr>
<td>LOG_INDEX_BUILDS</td>
<td></td>
<td>page 1-56</td>
</tr>
<tr>
<td>LOGSIZE</td>
<td></td>
<td>page 1-57</td>
</tr>
<tr>
<td>LTapeBLK</td>
<td></td>
<td>page 1-58</td>
</tr>
<tr>
<td>LTapeDEV</td>
<td></td>
<td>page 1-58</td>
</tr>
<tr>
<td>LTapeSIZE</td>
<td></td>
<td>page 1-59</td>
</tr>
<tr>
<td>LTxEHWM</td>
<td></td>
<td>page 1-59</td>
</tr>
<tr>
<td>LTxHWM</td>
<td></td>
<td>page 1-60</td>
</tr>
<tr>
<td>MAX_FILL_DATA_PAGES</td>
<td></td>
<td>page 1-61</td>
</tr>
<tr>
<td>MAX_INCOMPLETE_CONNECTIONS</td>
<td></td>
<td>page 1-62</td>
</tr>
<tr>
<td>MAX_PDQPRIORITY</td>
<td></td>
<td>page 1-62</td>
</tr>
<tr>
<td>MIRROR</td>
<td></td>
<td>page 1-63</td>
</tr>
<tr>
<td>MIRROROFFSET</td>
<td></td>
<td>page 1-64</td>
</tr>
<tr>
<td>MIRRORPATH</td>
<td></td>
<td>page 1-64</td>
</tr>
<tr>
<td>MSGPATH</td>
<td></td>
<td>page 1-65</td>
</tr>
<tr>
<td>MULTIPROCESSOR</td>
<td></td>
<td>page 1-65</td>
</tr>
<tr>
<td>NETTYPE</td>
<td></td>
<td>page 1-66</td>
</tr>
<tr>
<td>NOAGE</td>
<td></td>
<td>page D-7</td>
</tr>
<tr>
<td>NUMAIOVPS</td>
<td></td>
<td>page D-7</td>
</tr>
<tr>
<td>NUMCPUVPS</td>
<td></td>
<td>page D-8</td>
</tr>
<tr>
<td>OFF_RECVRY_THREADS</td>
<td></td>
<td>page 1-69</td>
</tr>
<tr>
<td>ON_RECVRY_THREADS</td>
<td></td>
<td>page 1-70</td>
</tr>
<tr>
<td>ONDBSPACEDOWN</td>
<td></td>
<td>page 1-71</td>
</tr>
<tr>
<td>ONLIDX_MAXMEM</td>
<td></td>
<td>page 1-72</td>
</tr>
<tr>
<td>OPCACHEMAX</td>
<td>INFORMIXOPCACHE</td>
<td>page 1-72</td>
</tr>
<tr>
<td>OPTCOMPIND</td>
<td>OPTCOMPIND</td>
<td>page 1-73</td>
</tr>
<tr>
<td>Configuration Parameter</td>
<td>Related Environment Variable</td>
<td>Reference</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>OPT_GOAL</td>
<td>OPT_GOAL</td>
<td>page 1-73</td>
</tr>
<tr>
<td>PC_HASHSIZE</td>
<td></td>
<td>page 1-74</td>
</tr>
<tr>
<td>PC_POOLSIZE</td>
<td></td>
<td>page 1-75</td>
</tr>
<tr>
<td>PHYSBUFF</td>
<td></td>
<td>page 1-75</td>
</tr>
<tr>
<td>PHYSDBS</td>
<td></td>
<td>page 1-76</td>
</tr>
<tr>
<td>PHYSFILE</td>
<td></td>
<td>page 1-76</td>
</tr>
<tr>
<td>QSTATS</td>
<td></td>
<td>page 1-77</td>
</tr>
<tr>
<td>RA_PAGES</td>
<td></td>
<td>page 1-77</td>
</tr>
<tr>
<td>RA_THRESHOLD</td>
<td></td>
<td>page 1-78</td>
</tr>
<tr>
<td>RESIDENT</td>
<td></td>
<td>page 1-78</td>
</tr>
<tr>
<td>RESTARTABLE_RESTORE</td>
<td></td>
<td>page 1-79</td>
</tr>
<tr>
<td>ROOTNAME</td>
<td></td>
<td>page 1-80</td>
</tr>
<tr>
<td>ROOTOFFSET</td>
<td></td>
<td>page 1-80</td>
</tr>
<tr>
<td>ROOTPATH</td>
<td></td>
<td>page 1-81</td>
</tr>
<tr>
<td>ROOTSIZE</td>
<td></td>
<td>page 1-81</td>
</tr>
<tr>
<td>RTO_SERVER_RESTART</td>
<td></td>
<td>page 1-82</td>
</tr>
<tr>
<td>SBSPACENAME</td>
<td></td>
<td>page 1-82</td>
</tr>
<tr>
<td>SBSPACETEMP</td>
<td></td>
<td>page 1-83</td>
</tr>
<tr>
<td>SDS_ENABLE</td>
<td></td>
<td>page 1-84</td>
</tr>
<tr>
<td>SDS_PAGING</td>
<td></td>
<td>page 1-84</td>
</tr>
<tr>
<td>SDS_TEMPPDBS</td>
<td></td>
<td>page 1-84</td>
</tr>
<tr>
<td>SDS_TIMEOUT</td>
<td></td>
<td>page 1-85</td>
</tr>
<tr>
<td>SECURITY_LOCALCONNECTION</td>
<td></td>
<td>page 1-86</td>
</tr>
<tr>
<td>SERVERNUM</td>
<td></td>
<td>page 1-86</td>
</tr>
<tr>
<td>SHMADD</td>
<td></td>
<td>page 1-86</td>
</tr>
<tr>
<td>SHMBASE</td>
<td></td>
<td>page 1-86</td>
</tr>
<tr>
<td>SHMTOTAL</td>
<td></td>
<td>page 1-87</td>
</tr>
<tr>
<td>SHMVIRT_ALLOCSEG</td>
<td></td>
<td>page 1-88</td>
</tr>
<tr>
<td>SHMVIRTSIZE</td>
<td></td>
<td>page 1-89</td>
</tr>
<tr>
<td>SINGLE_CPU_VP</td>
<td></td>
<td>page 1-90</td>
</tr>
<tr>
<td>SQLTRACE</td>
<td></td>
<td>page 1-91</td>
</tr>
<tr>
<td>STACKSIZE</td>
<td>INFORMIXSTACKSIZE</td>
<td>page 1-92</td>
</tr>
<tr>
<td>STAGEBLOB</td>
<td></td>
<td>page 1-92</td>
</tr>
<tr>
<td>STMT_CACHE</td>
<td>STMT_CACHE</td>
<td>page 1-92</td>
</tr>
<tr>
<td>STMT_CACHE_HITS</td>
<td></td>
<td>page 1-93</td>
</tr>
<tr>
<td>STMT_CACHE_NOLIMIT</td>
<td></td>
<td>page 1-94</td>
</tr>
<tr>
<td>STMT_CACHE_NUMPOOL</td>
<td></td>
<td>page 1-95</td>
</tr>
<tr>
<td>STMT_CACHE_SIZE</td>
<td></td>
<td>page 1-95</td>
</tr>
<tr>
<td>SYSALARMPROGRAM</td>
<td></td>
<td>page 1-95</td>
</tr>
<tr>
<td>SYSSBSPACENAME</td>
<td></td>
<td>page 1-96</td>
</tr>
<tr>
<td>TAPEBLK</td>
<td></td>
<td>page 1-97</td>
</tr>
<tr>
<td>Configuration Parameter</td>
<td>Related Environment Variable</td>
<td>Reference</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>TAPEDEV</td>
<td></td>
<td>page 1-98</td>
</tr>
<tr>
<td>TAPESIZE</td>
<td></td>
<td>page 1-99</td>
</tr>
<tr>
<td>TBLSPACE_STATS</td>
<td></td>
<td>page 1-99</td>
</tr>
<tr>
<td>TBLTBLFIRST</td>
<td></td>
<td>page 1-100</td>
</tr>
<tr>
<td>TBLTBLNEXT</td>
<td></td>
<td>page 1-100</td>
</tr>
<tr>
<td>TEMPTAB_NOLOG</td>
<td></td>
<td>page 1-100</td>
</tr>
<tr>
<td>TXTIMEOUT</td>
<td></td>
<td>page 1-101</td>
</tr>
<tr>
<td>USELASTCOMMITTED</td>
<td></td>
<td>page 1-101</td>
</tr>
<tr>
<td>USEOSTIME</td>
<td></td>
<td>page 1-102</td>
</tr>
<tr>
<td>UNSECURE_ONSTAT</td>
<td></td>
<td>page 1-101</td>
</tr>
<tr>
<td>VP_MEMORY_CACHE_KB</td>
<td></td>
<td>page 1-103</td>
</tr>
<tr>
<td>VPCLASS</td>
<td></td>
<td>page 1-103</td>
</tr>
<tr>
<td>WSTATS</td>
<td></td>
<td>page 1-108</td>
</tr>
</tbody>
</table>

### Parameter Attributes

This chapter describes one or more of the following attributes (if relevant) for each parameter.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>onconfig.std value</strong></td>
<td>The default value that appears in the onconfig.std file. The database server uses these default values for all configurations.</td>
</tr>
<tr>
<td><strong>if not present</strong></td>
<td>The value that the database server supplies if the parameter is missing from your ONCONFIG file. If this value is present in onconfig.std, the database server uses the onconfig.std value. If this value is not present in onconfig.std, the database server calculates the value based on other values in onconfig.std.</td>
</tr>
<tr>
<td><strong>units</strong></td>
<td>The units in which the parameter is expressed.</td>
</tr>
<tr>
<td><strong>separators</strong></td>
<td>The separators that can be used when the parameter value has several parts. Do not use white space within a parameter value.</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>The valid values for this parameter.</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>The time at which a change to the value of the parameter affects the operation of the database server. Disk is initialized means to reinitialize the database server.</td>
</tr>
<tr>
<td><strong>utilities</strong></td>
<td>The database server utilities that you can use to change the value of the parameter.</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>Cross-reference to further discussion.</td>
</tr>
</tbody>
</table>

### Using a Utility to Change a Parameter Value

Use one of these utilities to change the value of a configuration parameter. The utilities section for each configuration parameter lists the specific utilities to use.
<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON–Monitor (UNIX)</td>
<td>You can use ON–Monitor to change certain parameter values. In ON–Monitor, some of the responses are Y/N (yes/no). When those responses are recorded in the ONCONFIG file, Y becomes 1, and N becomes 0.</td>
</tr>
<tr>
<td>ISA</td>
<td>To use IBM Informix Server Administrator (ISA) to change parameter values, select Configuration &gt; ONCONFIG.</td>
</tr>
</tbody>
</table>

**Command-line utility**

The utilities section lists one or more command-line utilities that you can use to change a parameter value.

**Text editor**

You can use a text editor to modify the ONCONFIG file.

**Environment Variables**

If you set the environment variable on the database server, it applies to all sessions. If you set the environment variable in the client environment, it applies to the current session and overrides the equivalent configuration parameter (if any). For a complete list of environment variables, and how to set them, see the *IBM Informix Guide to SQL: Reference*.

**Note:** The INFORMIXDIR environment variable must always be set.

**Archecker Configuration Parameters**

The `ac_config.std` template contains the default `archecker` configuration parameters. Usually, you would not change these parameters. However, if you need to change these parameters, copy the `ac_config.std` template to the `AC_CONFIG` file. (The `AC_CONFIG` environment variable specifies the location of the `AC_CONFIG` file.) The `archecker` utility uses these parameters when it verifies a backup or performs a table-level restore. For information on these parameters, see the *IBM Informix Backup and Restore Guide*.

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC_DEBUG</td>
<td>Prints debugging messages in the <code>archecker</code> message log.</td>
</tr>
<tr>
<td>AC_IXBAR</td>
<td>Specifies the pathname to the IXBAR file.</td>
</tr>
<tr>
<td>AC_LTAPEBLOCK</td>
<td>Specifies the <code>ontape</code> block size for reading logical logs.</td>
</tr>
<tr>
<td>AC_LTAPEDEV</td>
<td>Specifies the local device name used by <code>ontape</code> for reading logical logs.</td>
</tr>
<tr>
<td>AC_MSGPATH</td>
<td>Specifies the location of the <code>archecker</code> message file.</td>
</tr>
<tr>
<td>AC_SCHEMA</td>
<td>Specifies the pathname to the <code>archecker schema</code> command.</td>
</tr>
<tr>
<td>AC_STORAGE</td>
<td>Specifies the location of the temporary files that <code>archecker</code> builds.</td>
</tr>
<tr>
<td>AC_TAPEBLOCK</td>
<td>Specifies the tape block size in kilobytes.</td>
</tr>
</tbody>
</table>
AC_TAPEDEV Specifies the device name used by the ontape utility.

AC_TIMEOUT Specifies the timeout value for ON-Bar and archecker processes if one of them exits prematurely.

AC_VERBOSE Specifies either verbose or quiet mode for archecker messages.

**ADMIN_MODE_USERS**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>range of values</td>
<td>comma-separated user names, such as: Karin, Sarah, Andrew, up to a string of 127 characters</td>
</tr>
<tr>
<td>utilities</td>
<td>oninit -U, onmode -j -U, onmode -wm, and onmode -wf,</td>
</tr>
<tr>
<td>refer to</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• “Initialize Shared Memory Only” on page 9-2</td>
</tr>
<tr>
<td></td>
<td>• “Changing the Database Server to Administration Mode with the -j Option” on page 11-14</td>
</tr>
<tr>
<td></td>
<td>• “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20</td>
</tr>
</tbody>
</table>

ADMIN_MODE_USERS specifies which users (in addition to user informix and DBSA group users) can access the database server in administration mode. The list of users is preserved indefinitely, and any or all of the users can be removed by using onmode -wm or onmode -wf.

Use onmode -j -U to allow any or more users to access the database server in administration mode when the database is running.

**ADMIN_USER_MODE_WITH_DBSA**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>range of values</td>
<td>0 = DBSA group users, user informix, and administration mode users as listed in ADMIN_MODE_USERS can connect 1 = only user informix can connect</td>
</tr>
<tr>
<td>refer to</td>
<td></td>
</tr>
<tr>
<td></td>
<td>“ADMIN_MODE_USERS” on page 1-13</td>
</tr>
</tbody>
</table>

ADMIN_USER_MODE_WITH_DBSA specifies whether user informix and the DBSA group users can connect to the database server while it is in administration mode.
ADTERR, ADTMODE, ADTPATH, and ADTSIZE (UNIX)

ADTERR, ADTMODE, ADTPATH, and ADTSIZE are configuration parameters for auditing. For information on these parameters, see the IBM Informix Security Guide.

ALARMPROGRAM

- onconfig.std value
  - On UNIX: /usr/informix/etc/no_log.sh
  - On Windows: %INFORMIXDIR%\etc\no_log.bat
- if not present
  - On UNIX: /usr/informix/etc/no_log.sh
  - On Windows: %INFORMIXDIR%\etc\no_log.bat
- range of values: Full pathname
- takes effect: When the database server is shut down and restarted
- refer to:
  - “Writing Your Own Alarm Script” on page C-1
  - IBM Informix Backup and Restore Guide

Use the ALARMPROGRAM parameter to display event alarms. The following sample scripts are provided.

<table>
<thead>
<tr>
<th>Script Name</th>
<th>Platform</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>log_full.sh</td>
<td>UNIX</td>
<td>To back up logical logs automatically when the database server issues a log-full event alarm, set ALARMPROGRAM to log_full.sh or log_full.bat.</td>
</tr>
<tr>
<td>log_full.bat</td>
<td>Windows</td>
<td></td>
</tr>
<tr>
<td>no_log.sh</td>
<td>UNIX</td>
<td>To disable automatic logical-log backups, set ALARMPROGRAM to no_log.sh or no_log.bat.</td>
</tr>
<tr>
<td>no_log.bat</td>
<td>Windows</td>
<td></td>
</tr>
<tr>
<td>alarmprogram.sh</td>
<td>UNIX</td>
<td>Handles event alarms and controls logical-log backups. Modify alarmprogram.sh or alarmprogram.bat and set ALARMPROGRAM to the full pathname of alarmprogram.sh or alarmprogram.bat. See “Customizing the ALARMPROGRAM scripts” on page C-1</td>
</tr>
<tr>
<td>alarmprogram.bat</td>
<td>Windows</td>
<td></td>
</tr>
</tbody>
</table>

Important: Backup media should always be available for automatic log backups.

You can set the ALRM_ALL_EVENTS configuration parameter to specify whether ALARMPROGRAM runs for all events that are logged in the MSGPATH or only for specified noteworthy events (events greater than severity 1).

Instead of using the supplied scripts, you can write your own shell script, batch file, or binary program to execute events. Set ALARMPROGRAM to the full pathname of this file. The database server executes this script when noteworthy events occur. These events include database, table, index, or simple-large-object failure; all logs are full; internal subsystem failure; initialization failure; and long transactions. You can have the events noted in an email or pagermail message.

ALLOW_NEWLINE

- onconfig.std value
  - 0
You can specify that you want the database server to allow the newline character (\n) in a quoted string either for all sessions or for a specific session. A session is the duration of a client connection to the database server.

To allow or disallow newline characters in quoted strings for all sessions, set the ALLOW_NEWLINE parameter in the ONCONFIG file. To allow all remote sessions in a distributed query to support embedded newline characters, specify ALLOW_NEWLINE in their ONCONFIG files.

To allow or disallow a newline character in a quoted string for a particular session when ALLOW_NEWLINE is not set, you must execute the ifx_allow_newline(boolean) user-defined routine (UDR).

**ALRM_ALL_EVENTS**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>range of values</td>
<td>0, 1</td>
</tr>
</tbody>
</table>

ALRM_ALL_EVENTS specifies whether ALARMPROGRAM runs for all events that are logged in the MSGPATH or only for noteworthy events. If ALRM_ALL_EVENTS is set to 1, it will trigger the ALARMPROGRAM and it will display all event alarms.

**AUTO_AIOVPS**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>range of values</td>
<td>0 = Off</td>
</tr>
<tr>
<td></td>
<td>1 = On</td>
</tr>
<tr>
<td>utilities</td>
<td>onmode -wf or onmode -wm</td>
</tr>
<tr>
<td>refer to</td>
<td>&quot;onmode -wf, -wm: Dynamically change certain configuration parameters&quot; on page 11-20</td>
</tr>
</tbody>
</table>

AUTO_AIOVPS enables the database server to automatically increase the number of AIO VPs and page cleaner threads when the database server detects that the I/O workload has outpaced the performance of the existing AIO VPs. You can
dynamically enable or disable the automatic increase of AIO VPs and page cleaner threads by using `onmode -wm` or `onmode -wf`.

### AUTO_CKPTS

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>
| range of values     | 0 = Off  
|                     | 1 = On |
| utilities           | `onmode -wf` or `onmode -wm` |
| refer to            | “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20 |

AUTO_CKPTS allows the server to trigger checkpoints more frequently to avoid transaction blocking. You can dynamically enable or disable automatic checkpoints by using `onmode -wm` or `onmode -wf`.

### AUTO_LRU_TUNING

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>
| range of values     | 0 = Off  
|                     | 1 = On |
| utilities           | `onmode -wf` or `onmode -wm` |
| refer to            | The following: |
|                     | • “onmode -wm: Change LRU tuning status” on page 11-21 |
|                     | • “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20 |

AUTO_LRU_TUNING enables automatic LRU tuning. You can dynamically enable or disable automatic LRU tuning by using `onmode -wm` or `onmode -wf`.

### AUTO_REPREPARE

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>
| range of values     | 0 = Disables the automatic repreparation of prepared objects after the schema of a directly or an indirectly referenced table is modified. Also disables the automatic reoptimization of SPL routines after the schema of an indirectly referenced table is modified.  
|                     | 1 = Enables the automatic repreparation and automatic reoptimization feature |
AUTO_REPREPARE controls whether a Dynamic Server feature is in effect that automatically re-optimizes SPL routines and re-prepares prepared objects after the schema of a table referenced by the SPL routine or by the prepared object has been changed.

When AUTO_REPREPARE is disabled, if DDL statements such as CREATE INDEX, DROP INDEX, DROP COLUMN, and RENAME COLUMN are executed, users of prepared objects that reference the modified tables or SPL routines that reference the modified tables indirectly receive -710 errors the next time they execute the prepared object or the SPL routine. Enabling AUTO_REPREPARE can avoid many -710 errors and can reduce the number of reprepare and reoptimize operations that users must perform manually after the schema of a table is modified.

**BLOCKTIMEOUT**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>3600</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Seconds</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

BLOCKTIMEOUT specifies the number of seconds that a thread or database server will hang. After the timeout, the thread or database server will either continue processing or fail.

**BTSCANNER**

**syntax**

```
BTSCANNER [num=scanner_threads] [,threshold=dirty_hits] 
[,rangesize=size] [,alice=mode]
```

| onconfig.std value | none |

<table>
<thead>
<tr>
<th>range of values</th>
</tr>
</thead>
<tbody>
<tr>
<td>num = The number of B-tree scanner threads to start at system startup. The default is 1.</td>
</tr>
<tr>
<td>threshold = The number of dirty hits (committed deleted index items) an index must encounter before the index is placed on the hot list for cleaning. Systems updated frequently should increase this value by a factor of 10x or 100x. The default is 5000.</td>
</tr>
<tr>
<td>rangesize = The size, in kilobytes, an index or index fragment must exceed before the index is cleaned with range scanning. To allow small indexes to be scanned by the leaf scan method set rangesize to 100. The default is OFF (-1).</td>
</tr>
<tr>
<td>alice = The mode for alice scanning. For small- to medium-sized systems with few or no indexes above 1 gigabyte, set alice to a mode of 6 or 7. For systems with large indexes, set alice to a higher mode.</td>
</tr>
</tbody>
</table>

The initial system-wide alice mode determines the initial size of the bitmaps that track the deleted index entries. Valid values range from 0 to 12. The default is OFF (0).
takes effect
When the database server is initialized. You can adjust these B-tree scanner settings with the onmode -C command while the database server is online.

refer to
See “onmode -C: Control the B-tree scanner” on page 11-6.

The BTSCANNER configuration parameter sets the B-tree scanner.

The B-tree scanner improves transaction processing for logged databases when rows are deleted from a table with indexes. The B-tree scanner threads remove deleted index entries and rebalance the index nodes. The B-tree scanner automatically determines which index items are to be deleted.

After all indexes above the threshold are cleaned, indexes below the threshold are added to the hot list. The default threshold is 500.

---

**BUFFERPOOL**

onconfig.std

values

---

**UNIX Only**

BUFFERPOOL default, lrus=8, buffers=5000, lru_min_dirty=50, lru_max_dirty=60
BUFFERPOOL size=2K, buffers=5000, lru=8, lru_min_dirty=50, lru_max_dirty=60

---

**End of UNIX Only**

---

**Windows Only**

BUFFERPOOL default, lrus=8, buffers=2000, lru_min_dirty=50, lru_max_dirty=60
BUFFERPOOL size=4K, buffers=2000, lru=8, lru_min_dirty=50, lru_max_dirty=60

---

**End of Windows Only**

---

**syntax**

BUFFERPOOL default, lrus=num_lrus, buffers=num_buffers,
lru_min_dirty=percent_min, lru_max_dirty=percent_max_dirty
BUFFERPOOL size=sizeK, buffers=num_buffers,
lrus=num_lrus, lru_min_dirty=percent_min,
lru_max_dirty=percent_max_dirty

---

**takes effect**
When the database server is shut down and restarted

---

**utilities**

onparams -b (See “onparams -b: Add a new buffer pool” on page 13-4.)
ontspaces (See “Specifying a Non-Default Page Size with the Same Size as the Buffer Pool” on page 14-11)
ON-Monitor (See Figure 12-7 on page 12-5)

---

**refer to**

“ontspaces -c -d: Create a dbspace” on page 14-8
The IBM Informix Dynamic Server Administrator’s Guide
The BUFFERPOOL configuration parameter specifies the default values for buffers and LRU queues in a buffer pool for both the default page size buffer pool and for any non-default pages size buffer pools.

**Note:** Information that was specified with the BUFFERS, LRU, LRU_MAX_DIRTY, and LRU_MIN_DIRTY configuration parameters prior to Version 10.0 is now specified using the BUFFERPOOL configuration parameter.

The BUFFERPOOL configuration parameter consists of two lines in the `onconfig.std` file, as shown in this example for a UNIX platform:

```plaintext
BUFFERPOOL default, lrus=8, buffers=5000, lru_min_dirty=50, lru_max_dirty=60
BUFFERPOOL size=2K, buffers=5000, lrus=8, lru_min_dirty=50, lru_max_dirty=60
```

The top line specifies the default values that are used if you create a dbspace with a page size that does not already have a corresponding buffer pool created at start up. The line below the default line specifies the database server’s default values for a buffer pool, which are based on the database server’s default page size. When you add a dbspace with a different page size with the onspaces utility or when you add a new buffer pool with the onparams utility, a new line is appended to the BUFFERPOOL configuration parameter in the ONCONFIG file. The page size for each buffer pool must be a multiple of the system’s default page size. Below is an example of the BUFFERPOOL lines where a third line has been appended:

```plaintext
BUFFERPOOL default, lrus=8, buffers=5000, lru_min_dirty=50, lru_max_dirty=60
BUFFERPOOL size=2K, buffers=5000, lrus=8, lru_min_dirty=50, lru_max_dirty=60
BUFFERPOOL size=6K, buffers=3000, lrus=8, lru_min_dirty=50, lru_max_dirty=60
```

The order of precedence for the BUFFERPOOL configuration parameter settings is:

1. The BUFFERPOOL size line, for example:
   - `BUFFERPOOL size=2K, buffers=5000, lrus=8, lru_min_dirty=50, lru_max_dirty=60`

2. Any deprecated parameters in the ONCONFIG file:
   - BUFFERS
   - LRU
   - LRU_MAX_DIRTY
   - LRU_MIN_DIRTY
   - For more information about deprecated configuration parameters, see [Appendix D, “Discontinued Configuration Parameters,” on page D-1.]

3. The BUFFERPOOL default line, for example:
   - `BUFFERPOOL default, lrus=8, buffers=5000, lru_min_dirty=50, lru_max_dirty=60`

4. Database server defaults.

When you use onspaces to create a new dbspace with a new page size, the database server takes the values of `buffers, lrus, lru_min_dirty` and `lru_max_dirty` from BUFFERPOOL default line unless there already is a BUFFERPOOL entry for that page size.

You can use the onparams utility when the database server is in online, quiescent, or in administration mode to add a new buffer pool with a different page size. There must be one buffer pool for each page size used by the dbspaces and all dbspaces using that page size must use the single buffer pool with that page size. When you use the onparams utility to add a buffer pool or when you add a dbspace with a different page size with the onspaces utility, the information you specify is automatically appended to the ONCONFIG file and new values are specified using the BUFFERPOOL keyword. You cannot change the values by
editing the **onconfig.std** file. If you need to resize or delete an existing buffer pool, you must restart the database server and then run **onparams** again.

Buffer pools that are added while the database server is running go into virtual memory, not into resident memory. Only those buffer pool entries that are specified in the ONCONFIG file at startup go into resident memory, depending on the availability of the memory you are using.

The fields in the BUFFERPOOL lines are not case sensitive (so you can specify **lrus** or **Lrus** or **LRUS**) and the fields can appear in any order.

For more information on buffer pools, including information on resizing and deleting buffer pools, see *IBM Informix Dynamic Server Administrator’s Guide*.

The following sections explain each of the fields in the BUFFERPOOL configuration parameter.

**The lrus Field**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>1rus=8</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>syntax</strong></td>
<td>1rus=num_lrus</td>
</tr>
<tr>
<td><strong>units</strong></td>
<td>Number of LRU queues</td>
</tr>
</tbody>
</table>
| **range of values**    | 32-bit platforms: 1 through 128  
                        | 64-bit platforms: 1 through 512 |

The **lrus** field specifies the number of LRU (least-recently-used) queues in the shared-memory buffer pool. You can tune the value of **lrus**, in combination with the **lru_min_dirty** and **lru_max_dirty** fields, to control how frequently the shared-memory buffers are flushed to disk.

Setting **lrus** too high might result in excessive page-cleaner activity.

**The buffers Field**

| **onconfig.std value** | UNIX: buffers=5000  
                        | Windows: buffers=2000 |
|------------------------|----------------------|
| **syntax**             | buffers=num_buffers |
| **units**              | Number of buffers. Each buffer is the size of the operating system page. |
| **range of values**    | For 32-bit platform on UNIX:  
                        | with page size equal to 2048 bytes:  
                        | 100 through 1,843,200 buffers  
                        | (1843200 = 1800 * 1024)  
                        | with page size equal to 4096 bytes:  
                        | 100 through 921,600 buffers  
                        | (921,600 = ((1800 * 1024)/4096) * 2048 )  
                        | For 32-bit platform on Windows:  
                        | 100 through 524,288 buffers (524,288 = 512 * 1024)  
                        | For 64-bit platforms: 100 through 2\(^{31}-1\) buffers  
                        | (For the actual value for your 64-bit platform, see your machine notes. The maximum number of buffers on Solaris is 536,870,912.) |
The `buffers` value specifies the maximum number of shared-memory buffers that the database server user threads have available for disk I/O on behalf of client applications. Therefore, the number of buffers that the database server requires depends on the applications. For example, if the database server accesses 15 percent of the application data 90 percent of the time, you need to allocate enough buffers to hold that 15 percent. Increasing the number of buffers can improve system performance.

**Recommendation:** Set the buffer space before you calculate other shared-memory parameters. On systems with a large amount of physical memory (4 GB or more), buffer space can be as much as 90 percent of physical memory.

**Buffers and Read-Ahead**

If you also want to perform read-ahead, increase the value of `buffers`. After you have configured all other shared-memory parameters, if you find that you can afford to increase the size of shared memory, increase the value of `buffers` until buffer space reaches the recommended 25 percent maximum.

**Buffers and Smart Large Objects**

If your databases contain smart large objects, you need to consider them when you calculate the value for `buffers`, because smart large objects are stored in the default page size buffer pool. If your applications frequently access smart large objects that are 2 kilobytes or 4 kilobytes in size, use the buffer pool to keep them in memory longer.

Use the following formula to increase the value of `buffers`:

\[ \text{Additional\_BUFFERS} = \text{numcur\_open\_lo} \times \left( \frac{\text{lo\_userdata}}{\text{pagesize}} \right) \]

- `numcur\_open\_lo` is the number of concurrently opened smart large objects that you can obtain from the `onstat -g smb fdd` option.
- `lo\_userdata` is the number of bytes of smart-large-object data that you want to buffer.
- `pagesize` is the page size in bytes for the database server.

As a general rule, try to have enough buffers to hold two smart-large-object pages for each concurrently open smart large object. (The additional page is available for read-ahead purposes).

If the system uses lightweight I/O (as set by the access-mode constant LO_NOBUFFER), the system allocates the buffers from shared memory and does not store the smart large objects in the buffer pool. For information on access-mode flags and constants, see the chapter on “Working with Smart Large Objects of the Universal Data Option” in the IBM Informix ESQL/C Programmer’s Manual.

**The lru_min_dirty Field**

- **onconfig.std value**
  
  lru_min_dirty=50

- **syntax**
  
  `lru_min_dirty=percent_min`

- **units**
  
  Percent

- **range of values**
  
  0 through 100 (fractional values are allowed)
The **lru_min_dirty** field specifies the percentage of modified pages in the LRU queues at which page cleaning is no longer mandatory. Page cleaners might continue cleaning beyond this point under some circumstances. If a field is specified out of the range of values, then the default of 80.00 percent is set.

**The lru_max_dirty Field**

```plaintext
onconfig.std value 1ru_max_dirty=60
syntax lru_max_dirty=percent_max
units Percent
range of values 0 through 100 (fractional values are allowed)
```

The **lru_max_dirty** field specifies the percentage of modified pages in the LRU queues at which the queue is cleaned. If a field is specified out of the range of values, then the default of 60.00 percent is set.

**The size Field**

```plaintext
onconfig.std value size=2K
syntax size=size
units Kilobytes
range of values 2 through 16
```

The **size** field specifies the page size for the particular BUFFERPOOL line. The K is optional.

**System Page Size**

The system page size is the default page size and is platform-dependent on Dynamic Server.

You can use the following utilities to display the system page size.

<table>
<thead>
<tr>
<th>Utility</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onstat -b</td>
<td>Displays the system page size, given as buffer size on the last line of the output</td>
</tr>
<tr>
<td>oncheck -pr</td>
<td>Checks the root-dbspace reserved pages and displays the system page size in the first section of its output</td>
</tr>
<tr>
<td>ON–Monitor (UNIX)</td>
<td>Displays the system page size under the Parameters &gt; Initialize option. Displays system page size under the Parameters &gt; Shared-Memory option, which does not require the database server to be running.</td>
</tr>
</tbody>
</table>

**CKPTINTVL**

```plaintext
onconfig.std value 300
units Seconds
range of values Any value greater than or equal to 0
```

1-22 IBM Informix Dynamic Server Administrator’s Reference
When the database server is shut down and restarted.

RTO_SERVER_RESTART and CKPTINTVL are mutually exclusive. If the RTO_SERVER_RESTART configuration parameter is enabled, it will trigger checkpoints and CKPTINTVL values are ignored. Otherwise, CKPTINTVL values are used to trigger checkpoints.

- Checkpoints, in the shared-memory and fast-recovery chapters of the *IBM Informix Administrator’s Guide*
- Your *IBM Informix Performance Guide*

CKPTINTVL specifies the frequency, expressed in seconds, at which the database server checks to determine whether a checkpoint is needed. When a checkpoint occurs, all pages in the shared-memory buffer pool are written to disk.

If you set CKPTINTVL to an interval that is too short, the system spends too much time performing checkpoints, and the performance of other work suffers. If you set CKPTINTVL to an interval that is too long, fast recovery might take too long.

In practice, 30 seconds is the smallest interval that the database server checks. If you specify a checkpoint interval of 0, the database server does not check if the checkpoint interval has elapsed. However, the database server still performs checkpoints. Other conditions, such as the physical log becoming 75 percent full, also cause the database server to perform checkpoints.

### CLEANERS

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Number of page-cleaner threads</td>
</tr>
<tr>
<td>range of values</td>
<td>1 through 128</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted.</td>
</tr>
<tr>
<td>utilities</td>
<td>onstat -F (see [15-19])</td>
</tr>
<tr>
<td>refer to</td>
<td>How the database server flushes data to disk, in the shared-memory chapter of the <em>IBM Informix Administrator’s Guide</em></td>
</tr>
</tbody>
</table>

CLEANERS specifies the number of page-cleaner threads available during the database server operation. By default, the database server always runs one page-cleaner thread. A general guideline is one page cleaner per disk drive. The value specified has no effect on the size of shared memory.

Based on the server work load, the server automatically attempts to optimize AIO VPs and page-cleaner threads and adjust the number of AIO VPs and page-cleaner threads upward when needed. Automatic AIO VP and page-cleaner thread tuning can be disabled using the environmental variable IFX_NO_AIOVP_TUNING or the onmode -wm utility option.
CONSOLE

onconfig.std value
On UNIX: /dev/console
On Windows: console.log

range of values
Pathname

takes effect
When the database server is shut down and restarted

refer to
The system console in the chapter on database server administration in the IBM Informix Administrator’s Guide

CONSOLE specifies the pathname and the filename for console messages.

DATASKIP

syntax
DATASKIP state [dbspace1 dbspace2 ...]
The state entry is required. If state is ON, at least one dbspace entry is required.

onconfig.std value
None

if not present
OFF

separators
Space

range of values
ALL = Skip all unavailable fragments.
OFF = Turn off DATASKIP.
ON = Skip some unavailable fragments.

utilities
onspaces -f (see 14-21)
onstat -f (see 15-18)

refer to
- “onspaces -f: Specify DATASKIP parameter” on page 14-21
- Your IBM Informix Performance Guide

DATASKIP lets you avoid points of media failure. This capability can result in higher availability for your data. To instruct the database server to skip some or all unavailable fragments, set this parameter. Whenever the database server skips over a dbspace during query processing, a warning is returned.

------------------------ ESQL/C

The previously reserved SQLCA warning flag sqlwarn.sqlwarn7 is set to W for IBM Informix ESQL/C

------------------------ End of ESQL/C

Use the following syntax in the parameter line:
DATASKIP OFF
DATASKIP ON dbspace1 dbspace2...
DATASKIP ALL

Use the -f option of the onspaces utility to alter the value of the DATASKIP parameter at runtime.
An application can use the SQL statement SET DATASKIP to override the DATASKIP value that the ONCONFIG parameter or onspaces sets. If the application then executes the SQL statement SET DATASKIP DEFAULT, the DATASKIP value for that session returns to whatever value is currently set for the database server.

**DBCREATE_PERMISSION**

<table>
<thead>
<tr>
<th>syntx</th>
<th>DBCREATE_PERMISSION value</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>Not set</td>
</tr>
<tr>
<td>units</td>
<td>usernames</td>
</tr>
<tr>
<td>separator</td>
<td>comma</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

DBCREATE_PERMISSION restricts the permission to create databases to the specified user. You can include multiple copies of DBCREATE_PERMISSION in the ONCONFIG file to give additional users permission to create databases.

The informix user always has permission to create databases. To restrict the ability to create databases to the informix user, add the following line to the ONCONFIG file:

```
DBCREATE_PERMISSION informix
```

**DB_LIBRARY_PATH**

<table>
<thead>
<tr>
<th>syntax</th>
<th>DB_LIBRARY_PATH value</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>None</td>
</tr>
<tr>
<td>if not present</td>
<td>The database server can load external modules from any location</td>
</tr>
<tr>
<td>range of values</td>
<td>List of path names (up to 512 bytes)</td>
</tr>
<tr>
<td>separators</td>
<td>Comma</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

The DB_LIBRARY_PATH configuration parameter specifies a comma-separated list of valid directory prefix locations from which the database server can load external modules, such as DataBlade™ Modules. You can also include server environment variables, such as $INFORMIXDIR.

You must specify paths for external modules exactly as they are registered with Dynamic Server. Relative paths or paths that include double periods (..) are not valid. External modules in the file systems that are not specified by this parameter cannot be loaded. This list is scanned prior to loading C language modules.

If you set this parameter, you must also include the string $INFORMIXDIR/extend as part of the value. If the string $INFORMIXDIR/extend is not included in DB_LIBRARY_PATH, IBM-supplied DataBlade Modules, the BladeManager, Large Object Locator DataBlade module functions, and DataBlade modules that you created with the DataBlade Developer's Kit will not load.
**DBSERVERALIASES**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>if not present</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>separators</strong></td>
<td>Comma</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>Up to 128 lowercase characters for each dbserver alias. Up to 32 values separated by commas. The value for DBSERVERALIASES follows the same rules as the DBSERVERNAME parameter (see “DBSERVERNAME” on page 1-27).</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When the database server is shut down and restarted. In addition, you might need to update the sqlhosts file or registry of each database server.</td>
</tr>
<tr>
<td><strong>MaxConnect users</strong></td>
<td>To use MaxConnect with more than one communication protocol, specify additional dbservernames in the DBSERVERALIASES parameter in the ONCONFIG file. The value of the INFORMIXSERVER environment variable on the client must match either the DBSERVERNAME or one of the entries of the DBSERVERALIASES parameter.</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>The following topics in the chapter on client/server communications in the IBM Informix Administrator’s Guide:</td>
</tr>
<tr>
<td></td>
<td>- ONCONFIG parameters for connectivity</td>
</tr>
<tr>
<td></td>
<td>- Using multiple connection types</td>
</tr>
</tbody>
</table>

DBSERVERALIASES specifies a list of alternative dbservernames and lets you assign multiple aliases to a database server, so each entry in the sqlhosts file or registry can have a unique name. If the database server supports more than one communication protocol (for example, both an IPC mechanism and the TCP network protocol), you must describe each valid connection to the database server with an entry in the sqlhosts file or registry. If the database server needs to support both SQLI and DRDA protocol (Client-Server communication protocol), you must assign an alias to the DRDA database server and add an entry in the sqlhosts file.

**Important:** You can specify up to 32 DBSERVERALIASES for a database server. If you attempt to define more than 32 DBSERVERALIASES, a warning message displays twice on the console. If you attempt to specify the DBSERVERALIASES all on one line, and the line exceeds 512 bytes, the excess bytes are truncated.

For each alternate name listed in DBSERVERALIASES, the database server starts an additional listener thread. If you have many client applications connecting to the database server, you can distribute the connection requests between several listener threads and reduce connection time. To take advantage of the alternate connections, instruct some of your client applications to use a CONNECT TO dbserveralias statement instead of CONNECT TO dbservername.
**DBSERVERNAME**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>On UNIX: hostname On Windows: ol_hostname (The hostname variable is the name of the host computer.)</td>
</tr>
</tbody>
</table>
| range of values    | Up to 128 lowercase characters DBSERVERNAME must begin with a letter and can include any printable character except the following characters:  
  - Uppercase characters  
  - A field delimiter (space or tab)  
  - A newline character  
  - A comment character  
  - A hyphen, minus, or @ character |
| takes effect       | When the database server is shut down and restarted. The sqlhosts file or registry of each database server that communicates with this database server might need to be updated. In addition, the INFORMIXSERVER environment variable for all users might need to be changed. |
| MaxConnect users   | The value of the INFORMIXSERVER environment variable on the client must match either the DBSERVERNAME or one of the entries of the DBSERVERALIASES parameter. |
| refer to           | DBSERVERNAME configuration parameter in the chapter on client/server communications in the IBM Informix Administrator's Guide |

When you install the database server, specify the dbservername. DBSERVERNAME specifies a unique name associated with this specific occurrence of the database server. The value of DBSERVERNAME is called the dbservername. Each dbservername is associated with a communication protocol in the sqlhosts file or registry. If the database server uses multiple communication protocols, additional values for dbservername must be defined with the DBSERVERALIASES configuration parameter.

Client applications use dbservername in the INFORMIXSERVER environment variable and in SQL statements such as CONNECT and DATABASE, which establish a connection to a database server.

**Important:** To avoid conflict with other instances of Informix database servers on the same computer or node, it is recommended that you use DBSERVERNAME to assign a dbservername explicitly.

**DBSPACETEMP**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>ROOTNAME</td>
</tr>
</tbody>
</table>
**separators**

Comma or colon (no white space)

**range of values**

The list of dbspaces can contain standard dbspaces, temporary dbspaces, or both. Use a colon or comma to separate the dbspaces in your list. The length of the list cannot exceed 254 characters.

**takes effect**

When the database server is shut down and restarted

**environment variable**

**DBSPACETEMP**

Specifies dbspaces that the database server uses to store temporary tables for a particular session. If **DBSPACETEMP** is not set, the default location is the root dbspace.

**utilities**

- **onspaces -t** (see [14-10])
- **onstat -d flags** field (see [15-14])

**refer to**

- What is a temporary table, in the chapter on data storage in the *IBM Informix Administrator’s Guide*
- *IBM Informix Guide to SQL: Reference*
- The order of precedence that the database server uses when it creates implicit sort files, in the *IBM Informix Performance Guide*
- The order of precedence of the default locations where the database server stores logged and unlogged temporary tables in the *IBM Informix Guide to SQL: Reference*.

DBSPACETEMP specifies a list of dbspaces that the database server uses to globally manage the storage of temporary tables. DBSPACETEMP improves performance by enabling the database server to spread out I/O for temporary tables efficiently across multiple disks. The database server also uses temporary dbspaces during backups to store the before-images of data that are overwritten while the backup is occurring.

DBSPACETEMP can contain dbspaces with a non-default page size, but all of the dbspaces in the DBSPACETEMP list must have the same page size. For more information about dbspaces in non-default buffer pools, see “BUFFERPOOL” on page 1-18.

If a client application needs to specify an alternative list of dbspaces to use for its temporary-table locations, the client can use the DBSPACETEMP environment variable to list them. The database server uses the storage locations that the DBSPACETEMP environment variable specifies only when you use the HIGH option of UPDATE STATISTICS.

**Important:** The dbspaces that you list in the DBSPACETEMP configuration parameter must consist of chunks that are allocated as raw UNIX devices. On Windows, you can create temporary dbspaces in NTFS files.

If both standard and temporary dbspaces are listed in the DBSPACETEMP configuration parameter or environment variable, the following rules apply:
• Sort, backup, implicit, and nonlogging explicit temporary tables are created in temporary dbspaces if adequate space exists.
• Explicit temporary tables created without the WITH NO LOG option are created in standard (rather than temporary) dbspaces.

When you create a temporary dbspace with ISA or with the onspaces utility, the database server does not use the newly created temporary dbspace until you perform the following steps.

To enable the database server to use the new temporary dbspace:
1. Add the name of a new temporary dbspace to your list of temporary dbspaces in the DBSPACETEMP configuration parameter, the DBSPACETEMP environment variable, or both.
2. Restart the database server with the oninit command (UNIX) or restart the database server service (Windows).

If you use the DBSPACETEMP environment variable to create a temporary dbspace in a user session, the change takes effect immediately and overrides the DBSPACETEMP value in the ONCONFIG file.

Using Hash Join Overflow and DBSPACETEMP
Dynamic Server uses an operating-system directory or file to direct any overflow that results from the following database operations if you do not set the DBSPACETEMP environment variable or DBSPACETEMP configuration parameter. You can specify the operating-system directory or file in the following ways:
• SELECT statement with GROUP BY clause
• SELECT statement with ORDER BY clause
• Hash-join operation
• Nested-loop join operation
• Index builds

If you do not set the DBSPACETEMP environment variable or DBSPACETEMP configuration parameter, the database server directs any overflow that results from the preceding operations to the operating-system directory or file that you specify in one of the following variables:

<table>
<thead>
<tr>
<th>UNIX Only</th>
</tr>
</thead>
<tbody>
<tr>
<td>• On UNIX, the operating-system directory or directories that the PSORT_DBTEMP environment variable specifies, if it is set. If PSORT_DBTEMP is not set, the database server writes sort files to the operating-system file space in the tmp directory.</td>
</tr>
</tbody>
</table>

| End of UNIX Only |

<table>
<thead>
<tr>
<th>Windows Only</th>
</tr>
</thead>
<tbody>
<tr>
<td>• On Windows, the directory specified in TEMP or TMP in the User Environment Variables window in Control Panel &gt; System.</td>
</tr>
</tbody>
</table>

| End of Windows Only |
**DD_HASHMAX**

- onconfig.std value: None
- units: Maximum number of tables in a hash bucket
- range of values: Positive integers
- takes effect: When the database server is shut down and restarted
- utilities: Use a text editor to modify the configuration file.
- refer to:
  - Configuration effects on memory, in your *IBM Informix Performance Guide*
  - "DD_HASHSIZE" on page 1-30

DD_HASHMAX specifies the maximum number of tables in each hash bucket in the data-dictionary cache. A *hash bucket* is the unit of storage (typically a page) whose address is computed by the hash function. A hash bucket contains several records.

For example, if DD_HASHMAX is 10 and DD_HASHSIZE is 100, you can store information about 1000 tables in the data-dictionary cache, and each hash bucket can have a maximum of 10 tables.

**DD_HASHSIZE**

- onconfig.std value: None
- units: Number of hash buckets or lists
- range of values: Any positive prime number
- takes effect: When the database server is shut down and restarted
- utilities: Use a text editor to modify the configuration file.
- refer to:
  - Configuration effects on memory, in your *IBM Informix Performance Guide*
  - "DD_HASHMAX" on page 1-30

DD_HASHSIZE specifies the number of hash buckets or lists in the data-dictionary cache.

**DEADLOCK_TIMEOUT**

- onconfig.std value: 60
- units: Seconds
- range of values: Positive integers
- takes effect: When the database server is shut down and restarted
- utilities: onstat -p dlout field (See 15-127)
- refer to: Configuration parameters used in two-phase
DEADLOCK_TIMEOUT specifies the maximum number of seconds that a database server thread can wait to acquire a lock. Use this parameter only for distributed queries that involve a remote database server. Do not use this parameter for nondistributed queries.

---

**DEF_TABLE_LOCKMODE**

<table>
<thead>
<tr>
<th>onconfig.std value if not present</th>
<th>PAGE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>PAGE = sets lock mode to page for new tables</td>
<td>R0W = sets lock mode to row for new tables</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
<td></td>
</tr>
<tr>
<td>environment variable</td>
<td>IFX_DEF_TABLE_LOCKMODE</td>
<td></td>
</tr>
<tr>
<td>refer to</td>
<td>• Environment variables in the <em>IBM Informix Guide to SQL: Reference</em></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Setting lock modes, in the <em>IBM Informix Guide to SQL: Tutorial</em></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Configuring lock mode, in the <em>IBM Informix Performance Guide</em></td>
<td></td>
</tr>
</tbody>
</table>

If DEF_TABLE_LOCKMODE = R0W, it sets the lock mode to row for every newly created table for all sessions that are connected to logging or nonlogging databases. This parameter has no effect on the lock mode for existing tables.

If DEF_TABLE_LOCKMODE is set to PAGE, the USELASTCOMMITTED configuration parameter and COMMITTED READ LAST COMMITTED option of the SET ISOLATION statement cannot enable access to the most recently committed data in tables on which uncommitted Read transactions hold shared locks, unless the tables were explicitly created or altered to have ROW as their locking granularity.

The rules of precedence for setting the lock mode are as follows.

<table>
<thead>
<tr>
<th>Precedence</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (highest)</td>
<td>CREATE TABLE or ALTER TABLE statements that use the LOCK MODE clause</td>
</tr>
<tr>
<td>2</td>
<td>IFX_DEF_TABLE_LOCKMODE environment variable set on the client side</td>
</tr>
<tr>
<td>3</td>
<td>IFX_DEF_TABLE_LOCKMODE environment variable set on the server side</td>
</tr>
<tr>
<td>4</td>
<td>DEF_TABLE_LOCKMODE value in ONCONFIG file</td>
</tr>
<tr>
<td>5 (lowest)</td>
<td>Default behavior (page-level locking)</td>
</tr>
</tbody>
</table>
DIRECT_IO (UNIX)

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
</table>
| range of values    | 0 = direct I/O is not used  
1 = direct I/O is used if available |
| takes effect       | When the database server is shut down and restarted |
| refer to           |  
- Direct I/O information in the *IBM Informix Performance Guide*  
- Direct I/O information in the *IBM Informix Administrator’s Guide*  
- “AUTO_AIOVPS” on page 1-15  
- “NUMAIOVPS” on page D-7 |

DIRECT_IO controls the use of direct I/O for cooked files used for dbspace chunks.

If you use direct I/O for cooked files used for dbspace chunks and the DIRECT_IO configuration parameter is enabled, you might be able to reduce the number of AIO virtual processors.

DIRECT_IO is not used for temporary dbspaces and can only be used for dbspace chunks whose file systems support direct I/O for the page size. If direct I/O is used for a dbspace chunk, KAIO (kernel asynchronous I/O) is used if supported by the file system. However, KAIO is not used if the environment variable KAIOFF is set. If DIRECT_IO is enabled and KAIO is also used, the number of AIO virtual processors can also be reduced. If DIRECT_IO is enabled and KAIO is not used, the number of AIO virtual processors should not be reduced.

**Note:**
Windows platforms do not recognize the value of the DIRECT I/O configuration parameter, and direct I/O is used for dbspace chunks on Windows® platforms regardless of the value of the DIRECT_I/O configuration parameter.

# DIRECTIVES

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>0 or 1</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>environment variable</td>
<td>IFX_DIRECTIVES</td>
</tr>
<tr>
<td>refer to</td>
<td></td>
</tr>
</tbody>
</table>
- Environment variables in the *IBM Informix Guide to SQL: Reference*  
- SQL directives, in the *IBM Informix Guide to SQL: Syntax*  
- Performance impact of directives, in your *IBM Informix Performance Guide* |
The DIRECTIVES parameter enables or disables the use of SQL directives. SQL directives allow you to specify behavior for the query optimizer in developing query plans for SELECT, UPDATE, and DELETE statements.

Set DIRECTIVES to 1, which is the default value, to enable the database server to process directives. Set DIRECTIVES to 0 to disable the database server from processing directives. Client programs also can set the IFX_DIRECTIVES environment variable to ON or OFF to enable or disable processing of directives by the database server. The setting of the IFX_DIRECTIVES environment variable overrides the setting of the DIRECTIVES configuration parameter. If you do not set the IFX_DIRECTIVES environment variable, all sessions for a client inherit the database server configuration for processing SQL directives.

**DISABLE_B162428_XA_FIX**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>None</td>
</tr>
<tr>
<td>units</td>
<td>Integer</td>
</tr>
<tr>
<td>range of values</td>
<td>0 = (Default) Frees transactions only when an xa_rollback is called</td>
</tr>
<tr>
<td></td>
<td>1 = Frees transactions if transaction rollback for other than an xa_rollback</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>refer to</td>
<td>IBM Informix Guide to SQL: Reference</td>
</tr>
</tbody>
</table>

Set DISABLE_B162428_XA_FIX to 1 to immediately free all global transactions after a transaction rollback, which is the default for Dynamic Server 9.40 and earlier versions. The default behavior for Dynamic Server 10.0 is to free global transactions after an xa_rollback is called, and this behavior is required to confirm to theXA state table that a transaction can be freed only after xa_rollback is called. Setting DISABLE_B162428_XA_FIX to 1 ensures that applications written for the earlier version of Dynamic server work properly.

You can override the DISABLE_B162428_XA_FIX configuration parameter for a client session with the IFX_XASTDCOMPLIANCE_XAEND environment variable. Setting IFX_XASTDCOMPLIANCE_XAEND to 1 will free transactions only when an xa_rollback is called. Setting IFX_XASTDCOMPLIANCE_XAEND to 0 will free transactions if the transaction rollback is for other than an xa_rollback.

**DRDA_COMMBUFFSIZE**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>32K</td>
</tr>
<tr>
<td>range of values</td>
<td>Minimum = 4 Kilobytes</td>
</tr>
<tr>
<td></td>
<td>Maximum = 2 Megabytes</td>
</tr>
<tr>
<td>takes effect</td>
<td>When shared memory is initialized</td>
</tr>
<tr>
<td>refer to</td>
<td>Setting the Size of the DRDA Communications Buffer in the IBM Informix Administrator’s Guide.</td>
</tr>
</tbody>
</table>

DRDA_COMMBUFFSIZE specifies the size of the DRDA communications buffer. When a DRDA session is established, the session is allocated a communication buffer equal to the current buffer size. If the buffer size is subsequently changed, existing connections are not affected, but new DRDA connections use the new size.
IDS silently resets values greater than 2 Megabyte to 2 Megabytes and resets values less than 4 Kilobytes to the 32 Kilobyte default value.

Users may specify the DRDA_COMMBUFFSIZE value in either MB or KB by adding either 'M' or 'K' to the value. The letter is case-insensitive, and the default is kilobytes. For example, a one megabyte buffer can be specified in any of these ways:
- DRDA_COMMBUFFSIZE 1M
- DRDA_COMMBUFFSIZE 1m
- DRDA_COMMBUFFSIZE 1024K
- DRDA_COMMBUFFSIZE 1024k
- DRDA_COMMBUFFSIZE 1024

**DRAUTO**

**onconfig.std value**

0

**range of values**

0 signifies OFF = Do not automatically switch the server type in the HDR environment.

1 signifies RETAIN_TYPE = Automatically switch secondary to standard during an HDR failure. Switch back to secondary when restarting HDR.

2 signifies REVERSE_TYPE = Automatically switch secondary to standard on an HDR failure. Switch to primary (and switch original primary to secondary) when restarting HDR.

**takes effect**

When shared memory is initialized

**utilities**

ON-Monitor > Parameters > data-Replication > Auto

**onstat** (See "onstat -g dri: Print HDR information" on page 15-38)

DRAUTO determines how a secondary database server reacts to an HDR failure. This parameter should have the same value on both HDR servers.

If DRAUTO is set to 0FF, the secondary database server remains a secondary database server in read-only mode when an HDR failure occurs.

If DRAUTO is set to either RETAIN_TYPE or REVERSE_TYPE, the secondary database server switches to type standard automatically when an HDR failure is detected. If DRAUTO is set to RETAIN_TYPE, the original secondary database server switches back to type secondary when the HDR connection is restored. If DRAUTO is set to REVERSE_TYPE, the original secondary database server switches to type primary when the HDR connection is restored, and the original primary switches to type secondary.

Use this parameter carefully. A network failure (that is, when the primary database server does not really fail, but the secondary database server perceives network slowness as an HDR failure) can cause the two database servers to become out of synch.
### DRIDXAUTO

**onconfig.std value**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Off</td>
</tr>
<tr>
<td>1</td>
<td>On</td>
</tr>
</tbody>
</table>

**range of values**

0 = Off
1 = On

**utilities**

`onstat` (See “onstat -g dri: Print HDR information” on page 15-38)

**takes effect**

When the database server is shut down and restarted

Specifies whether the primary High-Availability Data Replication (HDR) server automatically starts index replication if the secondary HDR server detects a corrupted index. To enable automatic index replication, set the value of the DRIDXAUTO configuration parameter to 1. You can alter the value of DRIDXAUTO for a running server instance without restarting the instance using the `onmode -d idxauto` command. However, the `onmode -d idxauto` command will not change the value of the DRIDXAUTO parameter in the ONCONFIG file.

For more information, see “onmode -d: Replicate an index with data-replication” on page 11-9.

### DRINTERVAL

**onconfig.std value**

30

**units**

Seconds

**range of values**

-1, 0, and positive integer values

**takes effect**

When the database server is shut down and restarted

**utilities**

`onstat` (See “onstat -g dri: Print HDR information” on page 15-38)

**refer to**

When log records are sent, in the chapter on High-Availability Data Replication in the IBM Informix Administrator’s Guide

DRINTERVAL specifies the maximum interval in seconds between flushing of the high-availability data-replication buffer. To update synchronously, set the parameter to -1.

### DRLOSTFOUND

**onconfig.std value**

On UNIX: `/usr/etc/dr.lostfound`
On Windows: `drive:\informix\etc\dr.lostfound`

**range of values**

Pathname

**takes effect**

When the database server is shut down and restarted

**utilities**

`onstat` (See “onstat -g dri: Print HDR information” on page 15-38)

**refer to**

Lost-and-found transactions, in the chapter on High-Availability Data Replication in the IBM Informix Administrator’s Guide
DRLOSTFOUND specifies the pathname to the `dr.lostfound.timestamp` file. This file contains transactions committed on the primary database server but not committed on the secondary database server when the primary database server experiences a failure. The file is created with a time stamp appended to the filename so that the database server does not overwrite another lost-and-found file if one already exists.

This parameter is not applicable if updating between the primary and secondary database servers occurs synchronously (that is, if DRINTERVAL is set to -1).

### DRTIMEOUT

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Seconds</td>
</tr>
<tr>
<td>range of values</td>
<td>Positive integers</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>utilities</td>
<td><code>onstat</code> (See <a href="on-page-15-38">onstat -g dri: Print HDR information</a>)</td>
</tr>
<tr>
<td>refer to</td>
<td>How High-Availability Data Replication failures are detected, in the chapter on High-Availability Data Replication in the <em>IBM Informix Administrator’s Guide</em></td>
</tr>
</tbody>
</table>

DRTIMEOUT applies only to high-availability data-replication pairs. This value specifies the length of time, in seconds, that a database server in a high-availability data-replication pair waits for a transfer acknowledgment from the other database server in the pair. Use the following formula to calculate DRTIMEOUT:

\[
\text{DRTIMEOUT} = \frac{\text{wait\_time}}{4}
\]

In this formula, `wait\_time` is the length of time, in seconds, that a database server in a high-availability data-replication pair must wait before it assumes that a high-availability data-replication failure occurred.

For example, suppose you determine that `wait\_time` for your system is 160 seconds. Use the preceding formula to set DRTIMEOUT as follows:

\[
\text{DRTIMEOUT} = \frac{160 \text{ seconds}}{4} = 40 \text{ seconds}
\]

### DS_HASHSIZE

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>31</td>
</tr>
<tr>
<td>units</td>
<td>Number of hash buckets or lists</td>
</tr>
<tr>
<td>range of values</td>
<td>Any positive prime number</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>refer to</td>
<td></td>
</tr>
</tbody>
</table>

- [IBM Informix Performance Guide](1-39) for how to monitor and tune the data-distribution cache
- [“DS_POOLSIZE” on page 1-39](1-39)
The DS_HASHSIZE parameter specifies the number of hash buckets in the data-distribution cache that the database server uses to store and access column statistics that the UPDATE STATISTICS statement generates in the MEDIUM or HIGH mode.

Use DS_HASHSIZE and DS_POOLSIZE to improve performance of frequently executed queries in a multiuser environment.

For information on configuration parameters for UDR cache, see “PC_HASHSIZE” on page 1-74 and “PC_POOLSIZE” on page 1-75.

**DS_MAX QUERIES**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>On UNIX: None On Windows: 32</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>num_cpu_vps * 2 * 128</td>
</tr>
<tr>
<td>units</td>
<td>Number of queries</td>
</tr>
<tr>
<td>range of values</td>
<td>Minimum = 1</td>
</tr>
<tr>
<td></td>
<td>Maximum = 8,388,608 (8 megabytes)</td>
</tr>
<tr>
<td>utilities</td>
<td>onmode -Q (see [11-10])</td>
</tr>
<tr>
<td></td>
<td>onstat -g mgm (See “onstat -g mgm: Print MGM resource information” on page 15-61)</td>
</tr>
<tr>
<td>refer to</td>
<td>“Specifying the Number of CPU VPs” on page 1-107</td>
</tr>
<tr>
<td></td>
<td>Parallel database query in your IBM Informix Performance Guide</td>
</tr>
</tbody>
</table>

DS_MAX QUERIES is the maximum number of PDQ queries that can run concurrently. The Memory Grant Manager (MGM) reserves memory for a query based on the following formula:

$$memory_{reserved} = DS\_TOTAL\_MEMORY \times \frac{(PDQ-priority / 100) \times (MAX\_PDQPRIORITY / 100)}{}$$

The value of PDQPRIORITY is specified in either the PDQPRIORITY environment variable or the SQL statement SET PDQPRIORITY.

**DS_MAX SCANS**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>1,048,576 or (1024 * 1024)</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Number of PDQ scan threads</td>
</tr>
<tr>
<td>range of values</td>
<td>10 through (1024 * 1024)</td>
</tr>
<tr>
<td>utilities</td>
<td>onmode -S (see [11-10])</td>
</tr>
<tr>
<td></td>
<td>onstat -g mgm (See “onstat -g mgm: Print MGM resource information” on page 15-61)</td>
</tr>
<tr>
<td>refer to</td>
<td>Parallel database query in your IBM Informix Performance Guide</td>
</tr>
</tbody>
</table>
DS_MAX_SCANS limits the number of PDQ scan threads that the database server can execute concurrently. When a user issues a query, the database server apportions some number of scan threads, depending on the following values:

- The value of PDQ priority (set by the environment variable PDQPRIORIY or the SQL statement SET PDQPRIORIY)
- The ceiling that you set with DS_MAX_SCANS
- The factor that you set with MAX_PDQPRIORIY
- The number of fragments in the table to scan (nfrags in the formula)

The Memory Grant Manager (MGM) tries to reserve scan threads for a query according to the following formula:

\[
\text{reserved_threads} = \min (nfrags, (DS\_MAX\_SCANS \times PDQPRIORIY / 100 \times \text{MAX\_PDQPRIORIY} / 100))
\]

If the DS_MAX_SCANS part of the formula is greater than or equal to the number of fragments in the table to scan, the query is held in the ready queue until as many scan threads are available as there are table fragments. Once underway, the query executes quickly because threads are scanning fragments in parallel.

For example, if nfrags equals 24, DS_MAX_SCANS equals 90, PDQPRIORIY equals 50, and MAX_PDQPRIORIY equals 60, the query does not begin execution until nfrags scan threads are available. Scanning takes place in parallel.

If the DS_MAX_SCANS formula falls below the number of fragments, the query might begin execution sooner, but the query takes longer to execute because some threads scan fragments serially.

If you reduce DS_MAX_SCANS to 40 in the previous example, the query needs fewer resources (12 scan threads) to begin execution, but each thread needs to scan two fragments serially. Execution takes longer.

**DS_NONPDQ_QUERY_MEM**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>128</td>
<td>Kilobytes</td>
</tr>
<tr>
<td>units</td>
<td></td>
<td>From 128 Kilobytes to 25 percent of the value of DS_TOTAL_MEMORY</td>
</tr>
<tr>
<td>range of values</td>
<td></td>
<td>When the database server is initialized</td>
</tr>
<tr>
<td>takes effect</td>
<td></td>
<td></td>
</tr>
<tr>
<td>utilities</td>
<td></td>
<td>onstat -g mgm (See <a href="#">onstat -g mgm: Print MGM resource information</a> on page 15-61)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>onmode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ON-Monitor</td>
</tr>
</tbody>
</table>

Use the DS_NONPDQ_QUERY_MEM configuration parameter to increase the amount of memory that is available for a query that is not a Parallel Database Query (PDQ). (You can only use this parameter if PDQ priority is set to zero.) If you specify a value for the DS_NONPDQ_QUERY_MEM parameter, determine and adjust the value based on the number and size of table rows.

The DS_NONPDQ_QUERY_MEM value is calculated during database server initialization based on the calculated DS_TOTAL_MEMORY value. If during the
processing of the DS_NONPDQ_QUERY_MEM, the database server changes the value that you set, the server sends a message in this format:

```
DS_NONPDQ_QUERY_MEM recalculated and changed from old_value Kb to new_value Kb.
```

In the message, old_value represents the value that you assigned to DS_NONPDQ_QUERY_MEM in the user configuration file, and new_value represents the value determined by the database server.

The value for DS_NONPDQ_QUERY_MEM can be changed using the onmode -wf option or superseded for a session with the onmode -wm option. For more information about onmode, see “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20.

**DS_POOLSIZE**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>127</td>
</tr>
<tr>
<td>units</td>
<td>Maximum number of entries in the data-distribution cache</td>
</tr>
<tr>
<td>range of values</td>
<td>Any positive value</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>
| refer to           | * IBM Informix Performance Guide for how to monitor and tune the data-distribution cache
|                     | * “DS_HASHSIZE” on page 1-36 |

The DS_POOLSIZE parameter specifies the maximum number of entries in each hash bucket in the data-distribution cache that the database server uses to store and access column statistics that the UPDATE STATISTICS statement generates in the MEDIUM or HIGH mode.

Use DS_HASHSIZE and DS_POOLSIZE to improve performance of frequently executed queries in a multi-user environment.

For information on configuration parameters for UDR cache, see “PC_HASHSIZE” on page 1-74 and “PC_POOLSIZE” on page 1-75.

**DS_TOTAL_MEMORY**

| onconfig.std value | On UNIX: None  
On Windows: 4,096 |
|--------------------|----------------|
| if not present     | If SHMTOTAL=0 and DS_MAX_QUERIES is set, DS_TOTAL_MEMORY = DS_MAX_QUERIES * 128.  
If SHMTOTAL=0 and DS_MAX_QUERIES is not set, DS_TOTAL_MEMORY = num_cpu_vps * 2 * 128. |
| units              | Kilobytes |
| range of values    | If DS_MAX_QUERY is set, the minimum value is DS_MAX_QUERY * 128. |
If DS_MAX_QUERY is not set, the minimum value is num_cpu_vps * 2 * 128.

Maximum value for 32-bit platform:
2 gigabytes
Maximum value for 64-bit platform:
4 gigabytes

utilities
onmode -M (see 11-10)
onstat -g mgm (See "onstat -g mgm: Print MGM resource information" on page 15-61)

refer to
- Your IBM Informix Performance Guide for the algorithms
- “SHMTOTAL” on page 1-88
- “SHMVIRTSIZE” on page 1-89
- “Specifying the Number of CPU VPs” on page 1-107
- The maximum memory available on your platform, in the machine notes

DS_TOTAL_MEMORY specifies the amount of memory available for PDQ queries. It should be smaller than the computer physical memory, minus fixed overhead such as operating-system size and buffer-pool size.

Do not confuse DS_TOTAL_MEMORY with the configuration parameters SHMTOTAL and SHMVIRTSIZE. SHMTOTAL specifies all the memory for the database server (total of the resident, virtual, and message portions of memory). SHMVIRTSIZE specifies the size of the virtual portion. DS_TOTAL_MEMORY is part of SHMVIRTSIZE.

For OLTP applications, set DS_TOTAL_MEMORY to between 20 and 50 percent of the value of SHMTOTAL in kilobytes.

For applications that involve large decision-support (DSS) queries, increase the value of DS_TOTAL_MEMORY to between 50 and 80 percent of SHMTOTAL. If you use your database server for DSS queries exclusively, set this parameter to 90 and 100 percent of SHMTOTAL.

Set the DS_TOTAL_MEMORY configuration parameter to any value not greater than the quantity (SHMVIRTSIZE - 10 megabytes).

**Algorithm for DS_TOTAL_MEMORY**
The database server derives a value for DS_TOTAL_MEMORY when you do not set DS_TOTAL_MEMORY, or if you set it to an inappropriate value. For information on the algorithms, see configuration effects on memory utilization in your IBM Informix Dynamic Server Performance Guide.

**DUMPCNT (UNIX)**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std value</td>
<td>1</td>
</tr>
<tr>
<td>if not present</td>
<td>1</td>
</tr>
<tr>
<td>units</td>
<td>Number of assertion failures</td>
</tr>
</tbody>
</table>
range of values Positive integers

takes effect When the database server is shut down and restarted

refer to Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator’s Guide*

DUMPCNT specifies the number of assertion failures for which one database server thread dumps shared memory or generates a core file by calling gcore. An assertion is a test of some condition or expression with the expectation that the outcome is true. For example, the following statement illustrates the concept of an assertion failure:

```c
if (a != b)
    assert_fail("a != b");
```

**DUMPCORE (UNIX)**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
</table>
| range of values    | 0 = Do not dump core image.  
|                    | 1 = Dump core image. |
| takes effect       | When the database server is shut down and restarted |
| refer to           | Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator’s Guide* |

DUMPCORE controls whether assertion failures cause a virtual processor to dump a core image. The core file is left in the directory from which the database server was last invoked. (The DUMPDIR parameter has no impact on the location of the core file.)

**Warning:** When DUMPCORE is set to 1, an assertion failure causes a virtual processor to dump a core image, which in turn causes the database server to abort. Set DUMPCORE only for debugging purposes in a controlled environment.

**DUMPDIR**

| onconfig.std value | On UNIX: /usr/informix/tmp  
|--------------------| On Windows: %INFORMIXDIR%/tmp |
| if not present     | %INFORMIXDIR%/tmp |
| range of values    | Any directory to which user informix has write access |
| takes effect       | When the database server is shut down and restarted |
| refer to           | Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator’s Guide* |

DUMPDIR specifies a directory in which the database server dumps shared memory, gcore files, or messages from a failed assertion. Because shared memory
can be large, set DUMPDIR to a file system with a significant amount of space. The directory to which DUMPDIR is set must exist for the server to start.

### DUMPGCORE (UNIX)

- **onconfig.std value**: 0
- **range of values**: 0 = Do not dump gcore. 1 = Dump gcore.
- **takes effect**: When the database server is shut down and restarted
- **refer to**: Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator’s Guide*

DUMPGCORE is used with operating systems that support gcore. If you set DUMPGCORE, but your operating system does not support gcore, messages in the database server message log indicate that an attempt was made to dump a core image, but the database server cannot find the expected file. (If your operating system does not support gcore, set DUMPCORE instead.)

If DUMPGCORE is set, the database server calls gcore whenever a virtual processor encounters an assertion failure. The gcore utility directs the virtual processor to dump a core image to the `core.pid.cnt` file in the directory that DUMPDIR specifies and continue processing.

The pid value is the process identification number of the virtual processor. The cnt value is incremented each time that this process encounters an assertion failure. The cnt value can range from 1 to the value of DUMPCNT. After that, no more core files are created. If the virtual processor continues to encounter assertion failures, errors are reported to the message log (and perhaps to the application), but no further diagnostic information is saved.

### DUMPShMEM (UNIX)

- **onconfig.std value**: 1
- **range of values**: 0 = Do not dump shared memory. 1 = Dump shared memory.
- **takes effect**: When the database server is shut down and restarted
- **refer to**: Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator’s Guide*

DUMPShMEM indicates that shared memory should be dumped on an assertion failure. All the shared memory that the database server uses is dumped; it is probably quite large. The shared-memory dump is placed in the `shm.pid.cnt` file in the directory that DUMPDIR specifies.

The pid value is the process identification number for the virtual processor. The cnt value is incremented each time that this virtual processor encounters an assertion failure. The cnt value can range from 1 to the value of DUMPCNT. After the value of DUMPCNT is reached, no more files are created. If the database server
continues to detect inconsistencies, errors are reported to the message log (and perhaps to the application), but no further diagnostic information is saved.

**DYNAMIC_LOGS**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None (this parameter is not in the onconfig.std file)</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>2 (Default)</td>
</tr>
<tr>
<td>range of values</td>
<td>0 = Turn off dynamic-log allocation.</td>
</tr>
<tr>
<td></td>
<td>1 = Set off the “log file required” alarm and pause to allow manual addition of a logical-log file. You can add a log file immediately after the current log file or to the end of the log file list.</td>
</tr>
<tr>
<td></td>
<td>2 = Turn on dynamic-log allocation. When the database server dynamically adds a log file, it sets off the “dynamically added log file” alarm.</td>
</tr>
<tr>
<td>takes effect</td>
<td>For HDR: when the database server is shut down and restarted</td>
</tr>
<tr>
<td></td>
<td>For Enterprise Replication: when Enterprise Replication is started</td>
</tr>
<tr>
<td>utilities</td>
<td>“onparams -a -d dbspace: Add a logical-log file” on page 13-2</td>
</tr>
<tr>
<td>refer to</td>
<td>• “LTXEHWM” on page 1-59</td>
</tr>
<tr>
<td></td>
<td>• “LTXHWM” on page 1-60</td>
</tr>
<tr>
<td></td>
<td>• Logical logs in the <em>IBM Informix Administrator’s Guide</em></td>
</tr>
</tbody>
</table>

If DYNAMIC_LOGS is 2, the database server automatically allocates a new log file when the next active log file contains an open transaction. Dynamic-log allocation prevents long transaction rollbacks from hanging the system.

If you want to choose the size and location of the new logical-log file, set DYNAMIC_LOGS to 1. Use the onparams -a command with the size (-s), location (-d dbspace), and -i options to add a log file after the current log file.

Even when DYNAMIC_LOGS is turned off, you do not have the same risks as in previous database server versions. In Version 9.3 and later, if the database server hangs from a long transaction rollback, you can shut down the database server, set DYNAMIC_LOGS to 1 or 2, and then restart the database server.

**Important:** If you are using Enterprise Replication with dynamic log allocation, set LTXEHWM to no higher than 70.

** ENCRYPT_CIPHERS **

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>allbut:&lt;ecb&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>syntax</td>
<td>ENCRYPT_CIPHERS all</td>
</tr>
<tr>
<td></td>
<td>• all</td>
</tr>
</tbody>
</table>

Chapter 1. Configuration Parameters  1-43
Specifies to include all available ciphers and modes, except ECB mode. For example: ENCRYPT_CIPHERS all

- allbut:<list of ciphers and modes>
  Specifies to include all ciphers and modes except the ones in the list. Separate ciphers or modes with a comma. For example: ENCRYPT_CIPHERS allbut:<cbc,bf>

- cipher:mode
  Specifies the ciphers and modes. Separate cipher-mode pairs with a comma. For example: ENCRYPT_CIPHERS des3:cbc,des3:ofb

_takes effect_

For HDR: when the database server is shut down and restarted

For Enterprise Replication: when Enterprise Replication is started

_refer to_

- "ENCRYPT_HDR" on page 1-45
- "ENCRYPT_MAC" on page 1-46
- "ENCRYPT_MACFILE" on page 1-46
- "ENCRYPT_SWITCH" on page 1-48
- HDR Encryption Options in the IBM Informix Administrator’s Guide
- Using High-Availability Data Replication in the IBM Informix Dynamic Server Enterprise Replication Guide

The ENCRYPT_CIPHERS configuration parameter defines all ciphers and modes that can be used by the current database session. ENCRYPT_CIPHERS is used for Enterprise Replication and High-Availability Data Replication only.

The cipher list for allbut can include unique, abbreviated entries. For example, bf can represent bf-1, bf-2, and bf-3; however, if the abbreviation is the name of an actual cipher, then only that cipher is eliminated. Therefore, des eliminates only the des cipher, but de eliminates des, des3, and desx.

**Important:** The encryption cipher and mode used is randomly chosen among the ciphers common between the two servers. It is strongly recommended that you do not specify specific ciphers. For security reasons, all ciphers should be allowed. If a specific cipher is discovered to have a weakness, then that cipher can be eliminated by using the allbut option.
The following ciphers are supported. For an updated list, see the Release Notes.

<table>
<thead>
<tr>
<th>Cipher</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>des</td>
<td>DES (64-bit key)</td>
</tr>
<tr>
<td>des3</td>
<td>Triple DES</td>
</tr>
<tr>
<td>desx</td>
<td>Extended DES (128-bit key)</td>
</tr>
<tr>
<td>aes</td>
<td>AES 128bit key</td>
</tr>
<tr>
<td>aes192</td>
<td>AES 192bit key</td>
</tr>
<tr>
<td>bf-1</td>
<td>Blow Fish (64-bit key)</td>
</tr>
<tr>
<td>bf-2</td>
<td>Blow Fish (128-bit key)</td>
</tr>
<tr>
<td>bf-3</td>
<td>Blow Fish (192-bit key)</td>
</tr>
<tr>
<td>aes128</td>
<td>AES 128bit key</td>
</tr>
<tr>
<td>aes256</td>
<td>AES 256bit key</td>
</tr>
</tbody>
</table>

The following modes are supported.
- **ecb**  Electronic Code Book (ECB)
- **cbc**  Cipher Block Chaining
- **cfb**  Cipher Feedback
- **ofb**  Output Feedback

All ciphers support all modes, except the `desx` cipher, which only supports the **cbc** mode.

Because **cdb** mode is considered weak, it is only included if specifically requested. It is not included in the **all** or the **allbut** list.

**ENCRIPT_HDR**

**onconfig.std** value 0

*range of values* 0 = Disables HDR encryption
1 = Enables HDR encryption

*takes effect* when the server is initialized

*refer to*
- “ENCRYPT_CIPHERS” on page 1-43
- “ENCRYPT_MAC” on page 1-46
- “ENCRYPT_MACFILE” on page 1-46
- “ENCRYPT_SWITCH” on page 1-48
- HDR Encryption Options in the *IBM Informix Administrator’s Guide*
- Using High-Availability Data Replication in the *IBM Informix Dynamic Server Enterprise Replication Guide*

**ENCRIPT_HDR** enables or disables HDR encryption. Enabling HDR encryption provides a secure method for transferring data from one server to another in an HDR pair. HDR encryption works in conjunction with Enterprise Replication (ER)
encryption. However, it is not necessary to have ER encryption enabled for HDR encryption. HDR encryption works whether ER encryption is enabled or not. HDR and ER share the same encryption configuration parameters: ENCRYPT_CIPHERS, ENCRYPT_MAC, ENCRYPT_MACFILE and ENCRYPT_SWITCH.

**ENCRYPT_MAC**

**onconfig.std value**

```
medium
```

**range of values**

One or more of the following options, separated by commas:

- **off** = does not use MAC generation
- **low** = uses XOR folding on all messages
- **medium** = uses SHA1 MAC generation for all messages greater than 20 bytes long and XOR folding on smaller messages
- **high** = uses SHA1 MAC generation on all messages

For example: ENCRYPT_MAC medium,high

**takes effect**

For HDR: when the database server is shut down and restarted
For Enterprise Replication: when Enterprise Replication is started

**refer to**

- “ENCRYPT_CIPHERS” on page 1-43
- “ENCRYPT_HDR” on page 1-45
- “ENCRYPT_MACFILE” on page 1-46
- “ENCRYPT_SWITCH” on page 1-48
- HDR Encryption Options in the *IBM Informix Administrator’s Guide*
- Using High-Availability Data Replication in the *IBM Informix Dynamic Server Enterprise Replication Guide*

The ENCRYPT_MAC configuration parameter controls the level of message authentication code (MAC) generation and is used for Enterprise Replication and High-Availability Data Replication only.

The level is prioritized to the highest value. For example, if one node has a level of **high** and **medium** enabled and the other node has only **low** enabled, then the connection attempt fails. Use the **off** entry between servers only when a secure network connection is guaranteed.

**ENCRYPT_MACFILE**

**onconfig.std value**

```
builtin
```

**units**

```
pathnames, up to 1536 bytes in length
```

**range of values**

One or more full path and filenames separated by commas, and the optional **builtin** keyword. For
example: ENCRYPT_MACFILE /usr/local/bin/mac1.dat, /usr/local/bin/mac2.dat,builtin

takes effect
For HDR: when the database server is shut down and restarted
For Enterprise Replication: when Enterprise Replication is started

refer to
• “ENCRYPT_CIPHERS” on page 1-43
• “ENCRYPT_HDR” on page 1-45
• “ENCRYPT_MAC” on page 1-46
• “ENCRYPT_SWITCH” on page 1-48
• HDR Encryption Options in the IBM Informix Administrator’s Guide
• Using High-Availability Data Replication in the IBM Informix Dynamic Server Enterprise Replication Guide

The ENCRYPT_MACFILE configuration parameter specifies a list of the full path names of MAC key files and is used for Enterprise Replication and High-Availability Data Replication only.

To specify the built-in key, use the keyword builtin. Using the builtin option provides limited message verification (some validation of the received message and determination that it appears to have come from a Dynamic Server client or server). The strongest verification is done by a site-generated MAC key file.

To generate a MAC key file:
1. Execute the following command from the command line:
   GenMacKey -o filename
   The filename is the name of the MAC key file.
2. Update the ENCRYPT_MACFILE configuration parameter on participating servers to include the location of the new MAC key file.
3. Distribute the new MAC key file.

Each of the entries for the ENCRYPT_MACFILE configuration parameter is prioritized and negotiated at connect time. The prioritization for the MAC key files is based on their creation time by the GenMacKey utility. The builtin option has the lowest priority. Because the MAC key files are negotiated, you should periodically change the keys.

<table>
<thead>
<tr>
<th>ENCRYPT_SMX</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std value</td>
<td>0</td>
</tr>
<tr>
<td>range of values</td>
<td>0, 1, 2</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the server is restarted</td>
</tr>
<tr>
<td>refer to</td>
<td>• HDR Encryption Options in the IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>
- Using High-Availability Data Replication in the IBM Informix Dynamic Server Enterprise Replication Guide

<table>
<thead>
<tr>
<th>Value</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (default)</td>
<td>Off. Do not encrypt.</td>
</tr>
<tr>
<td>1</td>
<td>On. Encrypt where possible.</td>
</tr>
<tr>
<td>2</td>
<td>On. Always encrypt.</td>
</tr>
</tbody>
</table>

The ENCRYPT_SMX configuration parameter sets the level of encryption for high-availability secondary server configurations. If the ENCRYPT_SMX parameter is set to 1 then encryption is used for SMX transactions only when the database server being connected to also supports encryption. If the ENCRYPT_SMX configuration parameter is set to 2 then only connections to encrypted database servers are allowed.

**ENCRIPT SWITCH**

**onconfig.std value**

60,60

**syntax**

ENCRIPT SWITCH cipher_switch_time, key_switch_time,

- cipher_switch_time specifies the minutes between cipher renegotiation
- key_switch_time specifies the minutes between secret key renegotiation

**units**

minutes

**range of values**

positive integers

**takes effect**

For HDR: when the database server is shut down and restarted

For Enterprise Replication: when Enterprise Replication is started

**refer to**

- "ENCRIPT_CIPHERS" on page 1-43
- "ENCRIPT_HDR" on page 1-45
- "ENCRIPT_MAC" on page 1-46
- "ENCRIPT_MACFILE" on page 1-46

HDR Encryption Options in the IBM Informix Administrator’s Guide

- Using High-Availability Data Replication in the IBM Informix Dynamic Server Enterprise Replication Guide

The ENCRYPT_SWITCH configuration parameter defines the frequency at which ciphers or secret keys are renegotiated. The longer the secret key and encryption cipher remains in use, the more likely the encryption rules might be broken by an attacker. To avoid this, cryptologists recommend changing the secret keys on long-term connections. The default time that this renegotiation occurs is once an hour. ENCRYPT_SWITCH is used for Enterprise Replication and High-Availability Data Replication only.
Enterprise Replication Configuration Parameters

The following configuration parameters apply to Enterprise Replication. For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

In addition, the ENCRYPT_CIPHERS, ENCRYPT_MAC, ENCRYPT_MACFILE and ENCRYPT_SWITCH configuration parameters apply to high-availability data replication (HDR). For more information, see the specific configuration parameter entries in this book.

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDR_DBSpace</td>
<td>Specifies the dbspace where the syscdr database is created.</td>
</tr>
<tr>
<td>CDR_DSLOCKWAIT</td>
<td>Specifies the number of seconds that the Datasync (data synchronization) component waits for database locks to be released.</td>
</tr>
<tr>
<td>CDR_ENV</td>
<td>Sets the Enterprise Replication environment variables CDR_LOGDELTA, CDR_PERFLOG, CRD_ROUTER, or CDR_RMSCALEFACT.</td>
</tr>
<tr>
<td>CDR_EVALTHREADS</td>
<td>Specifies the number of grouper evaluator threads to create when Enterprise Replication starts and enables parallelism.</td>
</tr>
<tr>
<td>CDR_MAX_DYNAMIC_LOGS</td>
<td>Specifies the number of dynamic log file requests that Enterprise Replication can make in one server session.</td>
</tr>
<tr>
<td>CDR_NIFCOMPRESS</td>
<td>Specifies the level of compression that the database server uses before sending data from the source database server to the target database server.</td>
</tr>
<tr>
<td>CDR_QDATA_SBSpace</td>
<td>Specifies the list of up to 32 names of sbspaces that Enterprise Replication uses to store spooled transaction row data.</td>
</tr>
<tr>
<td>CDR_QHDR_DBSpace</td>
<td>Specifies the location of the dbspace that Enterprise Replication uses to store the transaction record headers spooled from the send and receive queues.</td>
</tr>
<tr>
<td>CDR_QUEUEMEM</td>
<td>Specifies the maximum amount of memory that is used for the send and receive queues.</td>
</tr>
<tr>
<td>CDR_SERIAL</td>
<td>Controls generating values for SERIAL and SERIAL8 columns in tables defined for replication. Use this parameter to generate SERIAL column primary keys.</td>
</tr>
<tr>
<td>CDR_SUPPRESS_ATSRISWARN</td>
<td>Specifies the Datasync error and warning code numbers to be suppressed in the ATS and RIS files.</td>
</tr>
<tr>
<td>ENCRYPT_CDR</td>
<td>Specifies the level of Enterprise Replication encryption.</td>
</tr>
<tr>
<td>ENCRYPT_CIPHERS</td>
<td>Specifies the ciphers to use for Enterprise Replication encryption.</td>
</tr>
<tr>
<td>ENCRYPT_MAC</td>
<td>Specifies the level of message authentication coding to use with Enterprise Replication encryption.</td>
</tr>
</tbody>
</table>
ENCRYPT_MACFILE Specifies the message authentication coding key files to use with Enterprise Replication encryption.

ENCRYPT_SWITCH Defines the frequency at which ciphers and secret keys are re-negotiated for Enterprise Replication encryption.

### EXPLAIN_STAT

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>0 to 1</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

**refer to**

- "onmode -Y: Dynamically change SET EXPLAIN" on page 11-22
- SET EXPLAIN, in the *IBM Informix Guide to SQL: Syntax*
- Query Plan Report, in the *IBM Informix Performance Guide*

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Disables the display of query statistics</td>
</tr>
<tr>
<td>1</td>
<td>Enables the display of query statistics</td>
</tr>
</tbody>
</table>

The EXPLAIN_STAT configuration parameter enables or disables the inclusion of a Query Statistics section in the explain output file. You can generate the output file by using either the SET EXPLAIN statement of SQL or the `onmode -Y sessionid` command. When enabled, the Query Statistics section shows the Query Plan’s estimated number of rows, and the actual number of returned rows.

### EXT_DIRECTIVES

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>0, 1, 2</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

**environment variable** IFX_EXTDIRECTIVES

**refer to**

- Environment variables and information about the `sysdirectives` system catalog table, in the *IBM Informix Guide to SQL: Reference*
- SQL directives, in the *IBM Informix Guide to SQL: Syntax*
- Using external optimizer directives, in the *IBM Informix Performance Guide*

The EXT_DIRECTIVES configuration parameter enables or disables the use of external SQL directives. Enable external directives by using the EXT_DIRECTIVES configuration parameter in combination with the client-side IFX_EXTDIRECTIVES
environment variable as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (default)</td>
<td>Off. The directive cannot be enabled even if IFX_EXTDIRECTIVES is on.</td>
</tr>
<tr>
<td>1</td>
<td>On. The directive can be enabled for a session if IFX_EXTDIRECTIVES is on.</td>
</tr>
<tr>
<td>2</td>
<td>On. The directive can be used even if IFX_EXTDIRECTIVES is not set.</td>
</tr>
</tbody>
</table>

The setting of the IFX_EXTDIRECTIVES environment variable overrides the setting of the EXT_DIRECTIVES configuration parameter. If you do not set the IFX_EXTDIRECTIVES environment variable, all sessions for a client inherit the database server configuration for processing external directives.

**EXTSHMADD**

onconfig.std value 8192
range of values 1024 through 524,288
units Kilobytes
takes effect When the database server is shut down and restarted
utilities onstat -g seg

EXTSHMADD specifies the size of extension virtual segments that you add. Other virtual segment additions are based on the size that is specified in the SHMADD configuration parameter.

**FASTPOLL**

onconfig.std value 0
range of values 0 = Disables fast polling,
1 = Enables fast polling.
takes effect When the database server is shut down and restarted

Use the FASTPOLL configuration parameter to enable or disable fast polling of your network. FASTPOLL is a platform-specific configuration parameter.

**FILLFACTOR**

onconfig.std value 90
units Percent
range of values 1 through 100
takes effect When the index is built. Existing indexes are not changed. To use the new value, the indexes must be rebuilt.

refer to “Structure of B-Tree Index Pages” on page 4-16

FILLFACTOR specifies the degree of index-page fullness. A low value provides room for growth in the index. A high value compacts the index. If an index is full
(100 percent), any new inserts result in splitting nodes. You can also set the FILLFACTOR as an option on the CREATE INDEX statement. The setting on the CREATE INDEX statement overrides the ONCONFIG file value.

**HETERO_COMMIT**

<table>
<thead>
<tr>
<th><code>onconfig.std</code> value</th>
<th>0</th>
</tr>
</thead>
</table>
| range of values      | 1 = Enable heterogeneous commit.  
                        | 0 = Disable heterogeneous commit.  |
| `takes effect`       | When the database server is shut down and restarted |
| `refer to`           | - Heterogeneous commit protocol, in the chapter on multiphase commit protocols in the *IBM Informix Administrator’s Guide*  
                        - *IBM Informix Enterprise Gateway Manager User Manual* |

The HETERO_COMMIT configuration parameter specifies whether or not the database server is prepared to participate with IBM Informix Gateway products in heterogeneous commit transactions. Setting HETERO_COMMIT to 1 allows a single transaction to update one non-Informix database (accessed with any of the Gateway products) and one or more Informix databases.

If HETERO_COMMIT is 0, a single transaction can update databases as follows:

- One or more Informix databases and no non-Informix databases
- One non-Informix database and no Informix databases

You can read data from any number of Informix and non-Informix databases, regardless of the setting of HETERO_COMMIT.

**IFX_EXTEND_ROLE**

<table>
<thead>
<tr>
<th><code>onconfig.std</code> value</th>
<th>1</th>
</tr>
</thead>
</table>
| range of values      | 1 or 0n (default) = Enables the EXTEND role so that administrators can grant privileges to a user to create or drop a UDR that has the EXTERNAL clause.  
                        | 0 or 0ff = Disables the EXTEND role so that any user can register an external routine. |
| `refer to`           | Information on security for external routines in the *IBM Informix Security Guide* |

Your database server administrator (DBSA), by default user informix, uses the IFX_EXTEND_ROLE parameter to implement security measures that establish which users can register DataBlade modules or user-defined routines (UDRs). This prevents unauthorized users from registering external routines.

**IFX_FOLDVIEW**

| `onconfig.std` value | 1 |
range of values

0 or 0ff (default) = Disables view folding
1 or 0n = Enables view folding

refer to

Information on security for external routines in the
IBM Informix Administrator’s Guide

IFX_FOLDVIEW enables or disables view folding. For certain situations where a
view is involved in a query, view folding can significantly improve the
performance of the query. In these cases, views are folded into a parent query
instead of the query results being put into a temporary table.

The following types of queries can take advantage of view folding:

- Views that contain a UNION ALL clause and the parent query has a regular
  join, Informix join, ANSI join, or an ORDER BY clause
- Views with multiple table joins where the main query contains Informix or ANSI
type outer joins

A temporary table is created and view folding is not performed for the following
types of queries that perform a UNION ALL operation involving a view:

- The view has one of the following clauses: AGGREGATE, GROUP BY, ORDER
  BY, UNION, DISTINCT, or OUTER JOIN (either Informix or ANSI type).
- The parent query has a UNION or UNION ALL clause.

ISM_DATA_POOL and ISM_LOG_POOL

The ISM_DATA_POOL and ISM_LOG_POOL parameters control where IBM
Informix Storage Manager stores backed-up data and logical logs. For information
on these parameters, see the IBM Informix Backup and Restore Guide or the IBM
Informix Storage Manager Administrator’s Guide.

Java Configuration Parameters

The following configuration parameters allow you to use J/Foundation, which
incorporates an embedded Java™ virtual machine on the database server. For more
information on these parameters, see J/Foundation Developer’s Guide.

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| AFCRASH                 | When the 0x10 bit is on for AFCRASH, all the
  messages that the Java Virtual Machine generates
  are logged into the JVM_vpid file, where vpid is
  the process ID of the Java virtual processor. This
  file is stored in the directory where the JVPLOG
  file is stored. |
| JDKVERSION              | Version number of the Java Development Kit (JDK)
  or Java Runtime Environment (JRE) release |
| JVPDEBUG                | When set to 1, writes tracing messages to the
  JVPLOG file |
| JVPHOME                 | Directory where the classes of the IBM Informix
  JDBC Driver are installed |
<p>| JVPLOGFILE              | Absolute pathname for your Java VP log files |
| JVPPROPFILE             | Absolute pathname for the Java VP properties file |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JVPJAVAHOME</td>
<td>Directory where the Java Runtime Environment (JRE) for the database server is installed</td>
</tr>
<tr>
<td>JVMTHREAD</td>
<td>Thread package (green or native) to use for the JVM</td>
</tr>
<tr>
<td>JVPJAVAHOME</td>
<td>Path from JVPJAVAHOME to the location of the Java VM libraries</td>
</tr>
<tr>
<td>JVPCLASSPATH</td>
<td>Initial Java class path setting</td>
</tr>
<tr>
<td>JVPCLASSPATH</td>
<td>Number of Java virtual processors that the database server should start. (See “VPCLASS” on page 1-103)</td>
</tr>
</tbody>
</table>

**LISTEN_TIMEOUT**

- **onconfig.std value**: 10
- **Units**: Seconds
- **takes effect**: When the database server is stopped and restarted
- **utilities**: onmode -wf
  - onmode-wm

**LISTEN_TIMEOUT** specifies the number of seconds the server waits for a connection. It can be set to a lower number to guard against faulty connection requests that might indicate a Denial of Service attack. See also information about the MAX_INCOMPLETE_CONNECTIONS configuration parameter on page 1-62.

Depending on the machine capability of holding the threads (in number), you can configure MAX_INCOMPLETEgetConnections to a higher value and depending on the network traffic, you can set LISTEN_TIMEOUT to a lower value to reduce the chance that an attack can reach the maximum limit.

Both the LISTEN_TIMEOUT and the MAX_INCOMPLETE.getConnections configuration parameters can be changed using the onmode -wf option or superseded for a session with the onmode -wm option. For more information about onmode, see “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20.

**LOCKS**

- **onconfig.std value**: 2,000
- **units**: Number of locks in the internal lock table
- **range of values**: 2,000 through 8,000,000 for 32-bit database servers, 2,000 through 500,000,000 for 64-bit database servers
- **takes effect**: When the database server is shut down and restarted
- **utilities**: onstat -k (see 15-121)
• The memory and locking chapters in your *IBM Informix Performance Guide*
• The shared memory chapter in the *IBM Informix Administrator’s Guide*

LOCKS specifies the initial size of the lock table. The lock table holds an entry for each lock that a session uses. If the number of locks that sessions allocate exceeds the value of LOCKS, the database server increases the size of the lock table.

Although each additional lock takes up 120 bytes of resident shared memory, locks can become a resource drain if you have a limited amount of shared memory. In addition, the amount of storage occupied by a single lock depends upon the word-length of the platform. On Solaris, for example, if you set LOCKS to 1000000 (no commas), the database server allocates 44 megabytes of resident shared memory for locks. The storage requirements on other platforms may vary.

**Tip:** When you drop a database, a lock is acquired and held on each table in the database until the database is dropped. For more information on the DROP DATABASE statement, see the *IBM Informix Guide to SQL: Syntax.*

### LOGBUFF

<table>
<thead>
<tr>
<th><code>onconfig.std</code> value</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>units</code></td>
<td>Kilobytes</td>
</tr>
<tr>
<td><code>range of values</code></td>
<td>32 kilobytes through (32767 * page size / 1024) kilobytes</td>
</tr>
<tr>
<td><code>takes effect</code></td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td><code>utilities</code></td>
<td><code>onstat -l buffer</code> field, second section. See “onstat -l” Print physical and logical log information” on page 15-122</td>
</tr>
<tr>
<td><code>refer to</code></td>
<td>Logical-log buffer, in the shared-memory chapter of the <em>IBM Informix Administrator’s Guide</em></td>
</tr>
</tbody>
</table>

LOGBUFF specifies the size in kilobytes for the three logical-log buffers in shared memory. Triple buffering permits user threads to write to the active buffer while one of the other buffers is being flushed to disk. If flushing is not complete by the time the active buffer fills, the user thread begins writing to the third buffer.

If you are using RTO_SERVER_RESTART, a LOGBUFF value of 256 kilobytes is recommended. If the LOGBUFF value is less than 256 kilobytes, a warning message displays when you restart the server. Otherwise, set LOGBUFF to 32 kilobytes for standard workloads or 64 kilobytes for heavy workloads. Choose a value for LOGBUFF that is evenly divisible by the page size. If the value of LOGBUFF is not evenly divisible by the page size, the database server rounds down the size to the nearest value that is evenly divisible by the page size.

If you log user data in smart large objects, increase the size of the log buffer to make the system more efficient. The database server logs only the portion of a smart-large-object page that changed.
**Important:** The database server uses the LOGBUFF parameter to set the size of internal buffers that are used during recovery. If you set LOGBUFF too high, the database server can run out of memory and shut down during recovery.

To set the system page size, use one of the utilities listed in “System Page Size” on page 1-22.

**LOGFILES**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>if not present</strong></td>
<td>6</td>
</tr>
<tr>
<td><strong>units</strong></td>
<td>Number of logical-log files</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>3 through 32,767 (integers only)</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>During disk initialization and when you add a new log file. You add a new log with one of the following utilities.</td>
</tr>
<tr>
<td><strong>utilities</strong></td>
<td>onparams (see 13-1)</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>The following topics in the IBM Informix Administrator’s Guide:</td>
</tr>
<tr>
<td></td>
<td>• Size of logical-log files, in the chapter on the logical log</td>
</tr>
<tr>
<td></td>
<td>• Adding or dropping a logical-log file, in the chapter on managing the logical log</td>
</tr>
</tbody>
</table>

LOGFILES specifies the number of logical-log files that the database server creates during disk initialization. To change the number of logical-log files, add or drop logical-log files.

If you use ISA or onparams to add or drop log files, the database server automatically updates LOGFILES.

**LOG_INDEX_BUILDS**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>0 (disabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>if not present</strong></td>
<td>0 (disabled)</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>0, 1 (0 = disable, 1 = enable)</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When the database server is stopped and restarted</td>
</tr>
<tr>
<td><strong>utilities</strong></td>
<td>onmode -wf</td>
</tr>
<tr>
<td></td>
<td>onmode -wm</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>The following topics in the IBM Informix Administrator’s Reference:</td>
</tr>
<tr>
<td></td>
<td>• 11-20</td>
</tr>
<tr>
<td></td>
<td>• Chapter 11, “The onmode Utility,” on page 11-1</td>
</tr>
</tbody>
</table>

LOG_INDEX_BUILDS is used to enable or disable index page logging. If LOG_INDEX_BUILDS is enabled, logical log file space consumption will increase,
depending on the size of the indexes. This might lead to logical log file backups being required more frequently. Messages are written to the online.log file when index page logging status changes.

Using `onmode -wm` enables or disables index page logging for the current session only, and does not affect the setting in the onconfig file. If the server is stopped and restarted, the setting in the onconfig file determines whether index page logging is enabled. Therefore, enabling index page logging using `onmode -wm` is not recommended when using RS secondary servers; instead, use `onmode -wf` to update the onconfig file, so that index page logging is enabled after restarting the server. Index page logging is a requirement when using RS secondary servers.

For more information about `onmode`, see “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20.

### LOGSIZE

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>2000</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>if not present</strong></td>
<td>UNIX: 1500 Windows: 500</td>
</tr>
<tr>
<td><strong>units</strong></td>
<td>Kilobytes</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>Minimum = 200 Maximum = ((\text{ROOTSIZE} - \text{PHYSFILE} - 512 - (63 \times ((\text{pagesize})/1024))) / \text{LOGFILES})</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When the database server is shut down and restarted. The size of log files added after shared memory is initialized reflects the new value, but the size of existing log files does not change.</td>
</tr>
<tr>
<td><strong>utilities</strong></td>
<td><code>onparams</code>&lt;br&gt;See &quot;onparams -p: Change physical-log parameters” on page 13-3</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>The following topics in the <em>IBM Informix Administrator’s Guide</em>:&lt;br&gt;• Size of the logical log and logging smart large objects, in the chapter on the logical log&lt;br&gt;• Changes to LOGSIZE or LOGFILES, in the chapter on managing logical logs&lt;br&gt;• “LTXHWM” on page 1-60</td>
</tr>
</tbody>
</table>

LOGSIZE specifies the size that is used when logical-log files are created. It does not change the size of existing logical-log files. The total logical-log size is \(\text{LOGSIZE} \times \text{LOGFILES} \).

To verify the page size that the database server uses on your platform, use one of the utilities listed in “System Page Size” on page 1-22.

### LOGSIZE for Smart Large Objects

If you declare logging for a smart-large-object column, you must ensure that the logical log is considerably larger than the amount of data logged during inserts or updates.
Important: The database server cannot back up open transactions. If many transactions are active, the total logging activity should not force open transactions to the log backup files. For example, if your log size is 1000 kilobytes and the high-watermark is 60 percent, do not use more than 600 kilobytes of the logical log for the smart-large-object updates. The database server starts rolling back the transaction when it reaches the high-watermark of 600 kilobytes.

LTAPEBLK

onconfig.std value 32
units Kilobytes
range of values Values greater than (page size/1024)

takes effect For ontape: When you execute ontape
For onload and onunload: When the database server is shut down and restarted

refer to

• Using ontape, in the IBM Informix Backup and Restore Guide
• Using onload and onunload, in the IBM Informix Migration Guide
• "TAPEBLK" on page 1-97

LTAPEBLK specifies the block size of the device to which the logical logs are backed up when you use ontape for dbspace backups. LTAPEBLK also specifies the block size for the device to which data is loaded or unloaded when you use the -l option of onload or onunload. If you are using onload or onunload, you can specify a different block size at the command line.

Specify LTAPEBLK as the largest block size permitted by your tape device. The database server does not check the tape device when you specify the block size. Verify that the LTAPEDEV tape device can read the block size that you specify. If not, you might not be able to read from the tape.

---

UNIX Only

The UNIX dd utility can verify that the LTAPEDEV tape device can read the block size. It is available with most UNIX systems.

---

End of UNIX Only

LTAPEDEV

onconfig.std value On UNIX: /dev/tapedev
On Windows: \TAPE1
if not present On UNIX: /dev/null
On Windows: nul
takes effect For ontape: when the database server is shut down
and restarted, if set to /dev/null on UNIX or nul on Windows. When you execute ontape, if set to a tape device.

For onload and onunload: when the database server is shut down and restarted

* How to set and change the LTAPEDEV value for ontape and how LTAPEDEV affects ON–Bar, in the IBM Informix Backup and Restore Guide
* Using onload or onunload, in the IBM Informix Migration Guide
* "TAPESIZE" on page 1-98

LTAPEDEV specifies the device or directory file system to which the logical logs are backed up when you use ontape for backups.

LTAPEDEV also specifies the device to which data is loaded or unloaded when you use the -l option of onload or onunload. If you are using LTAPEDEV to specify a device for onunload or onload, the same information for TAPEDEV is relevant for LTAPEDEV.

**Warning:** Do not set LTAPEDEV to /dev/null or nul when you use ON–Bar to back up logical logs.

### LTAPEDEV

| onconfig.std value | 10,240 |
| units              | Kilobytes |
| range of values    | 0 through 2,097,151 |
| takes effect       | For ontape: when you execute ontape For onload and onunload: when the database server is shut down and restarted |

**Warning:** Do not set LTAPEDEV to /dev/null or nul when you use ON–Bar to back up logical logs.

### LTAPESIZE

LTAPESIZE specifies the maximum tape size of the device to which the logical logs are backed up when you use ontape for backups. LTAPESIZE also specifies the maximum tape size of the device to which data is loaded or unloaded when you use the -l option of onload or onunload. If you are using onload or onunload, you can specify a different tape size on the command line. If you want to use the full capacity of a tape, set LTAPESIZE to 0.

### LTXEHWM

| onconfig.std value | None (not present in onconfig.std) |
if not present  
90 (if DYNAMIC_LOGS is set to 1 or 2)  
60 (if DYNAMIC_LOGS is set to 0)  

units  
Percent  

range of values  
LTXHWM through 100  

takes effect  
When the database server is shut down and restarted  

refer to  
• “DYNAMIC_LOGS” on page 1-43  
• “LTXHWM” on page 1-60  
• Setting high-watermarks for rolling back long transactions, in the chapter on managing logical logs in the IBM Informix Administrator’s Guide

A **transaction is long** if it is not committed or rolled back when it reaches the long-transaction high-watermark. LTXHWM specifies the **long-transaction, exclusive-access, high-watermark**. When the logical-log space reaches the LTXHWM threshold, the long transaction currently being rolled back is given exclusive access to the logical log.

If your system runs out of log space before the rollback completes, lower the LTXHWM value.

If you do not want too many logical logs to be added, LTXHWM should be set to a smaller value (around 60). If dynamic logging is turned off (DYNAMIC_LOGS = 0), LTXHWM should be set lower (around 50) to avoid running out of logical space.

**Tip:** To allow users to continue to access the logical logs, even during a long transaction rollback, set LTXHWM to 100. Set DYNAMIC_LOGS to 1 or 2 so that the database server can add log files as needed to complete the transaction or rollback.

---

**LTXHWM**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>None (not present in onconfig.std)</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>80 (if DYNAMIC_LOGS is set to 1 or 2)</td>
</tr>
<tr>
<td></td>
<td>50 (if DYNAMIC_LOGS is set to 0)</td>
</tr>
<tr>
<td>units</td>
<td>Percent</td>
</tr>
<tr>
<td>range of values</td>
<td>1 through 100</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

refer to  
• “DYNAMIC_LOGS” on page 1-43  
• “LTXHWM” on page 1-59  
• Setting high-watermarks for rolling back long transactions, in the chapter on managing logical logs in the IBM Informix Administrator’s Guide

LTXHWM specifies the long-transaction high-watermark. The **long-transaction high-watermark** is the percentage of available log space that, when filled, triggers...
the database server to check for a long transaction. When the logical-log space reaches the LTXHWM threshold, the database server starts rolling back the transaction. If you decrease the LTXHWM value, increase the size or number of log files to make rollbacks less likely.

If DYNAMIC_LOGS is set to 1 or 2, the database server adds as many logs are needed to complete the rollback.

If you do not want too many logical logs to be added, LTXHWM should be set to a smaller value (around 60). If dynamic logging is turned off (DYNAMIC_LOGS = 0), LTXHWM should be set lower (around 50) to avoid running out of logical space.

**Warning:** If you set both LTXHWM and LTXEHWM to 100, long transactions are never aborted. Although you can use this configuration to your advantage, you should set LTXHWM to below 100 for normal database server operations.

If you set LTXHWM to 100, the database server issues a warning message:

LTXHWM is set to 100. This long transaction high water mark will never be reached. Transactions will not be aborted automatically by the server, regardless of their length.

If the transaction hangs, follow the instructions for recovering from a long transaction hang, in the chapter on managing logical-log files in the *IBM Informix Administrator’s Guide*.

### MAX_FILL_DATA_PAGES

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>units</strong></td>
<td>Integer</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>0 or 1</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When the database server is stopped and restarted</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>Reducing disk space through additional rows per page in tables with variable-length rows in the <em>IBM Informix Performance Guide</em></td>
</tr>
</tbody>
</table>

Set the MAX_FILL_DATA_PAGES value to 1 to allow more rows to be inserted per page in tables that have variable-length rows. This setting can reduce disk space, make more efficient use of the buffer pool, and reduce table scan times.

If MAX_FILL_DATA_PAGES is enabled, the server will add a new row to a recently modified page with existing rows if adding the row leaves at least 10 percent of the page free for future expansion of all the rows in the page. If MAX_FILL_DATA_PAGES is not set, the server will add the row only if there is sufficient room on the page to allow the new row to grow to its maximum length.

A possible disadvantage of enabling MAX_FILL_DATA_PAGES and allowing more variable-length rows per page is that the server might store rows in a different physical order. Also, as the page fills, updates made to the variable-length columns in a row could cause the row to expand so it no longer completely fits on the page. This causes the server to split the row onto two pages, increasing the access time for the row.
To take advantage of this setting, existing tables with variable-length rows must be reloaded or existing pages must modified, followed by further inserts.

**MAX_INCOMPLETE_CONNECTIONS**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>1024</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Number of listener threads</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is stopped and restarted</td>
</tr>
<tr>
<td>utilities</td>
<td>onmode -wf</td>
</tr>
<tr>
<td></td>
<td>onmode-wm</td>
</tr>
<tr>
<td>refer to</td>
<td>IBM Informix Security Guide</td>
</tr>
</tbody>
</table>

Use MAX_INCOMPLETE_CONNECTIONS to specify the maximum number of incomplete connections in a session. After this number is reached, an error message is written in the online message log stating that the server might be under a Denial of Service attack. See also information about the LISTEN_TIMEOUT configuration parameter on page 1-54.

Depending on the machine capability of holding the threads (in number), you can configure MAX_INCOMPLETE_CONNECTIONS to a higher value and depending on the network traffic, you can set LISTEN_TIMEOUT to a lower value to reduce the chance that an attack can reach the maximum limit.

Both the MAX_INCOMPLETE_CONNECTIONS and the LISTEN_TIMEOUT configuration parameters can be changed using the onmode -wf option or superseded for a session with the onmode -wm option. For more information about onmode, see “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20.

**MAX_PDQPRIORITY**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>100</td>
</tr>
<tr>
<td>range of values</td>
<td>0 through 100</td>
</tr>
<tr>
<td>takes effect</td>
<td>On all user sessions</td>
</tr>
<tr>
<td>utilities</td>
<td>onmode -D</td>
</tr>
<tr>
<td>onstat -g mgm</td>
<td>(See “onstat -g mgm: Print MGM resource information” on page 15-61)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>refer to</th>
</tr>
</thead>
<tbody>
<tr>
<td>• The chapter on using PDQ, in the <strong>IBM Informix Performance Guide</strong></td>
</tr>
<tr>
<td>• “onmode -D, -M, -Q, -S: Change decision-support parameters” on page 11-10</td>
</tr>
</tbody>
</table>

MAX_PDQPRIORITY limits the PDQ resources that the database server can allocate to any one DSS query. MAX_PDQPRIORITY is a factor that is used to scale the value of PDQ priority set by users. For example, suppose that the database administrator sets MAX_PDQPRIORITY to 80. If a user sets the PDQPRIORITY environment variable to 50 and then issues a query, the database server silently processes the query with a PDQ priority of 40.
You can use the `onmode` utility to change the value of MAX_PDQPRIORITY while the database server is online.

In Dynamic Server, PDQ resources include memory, CPU, disk I/O, and scan threads. MAX_PDQPRIORITY lets the database administrator run decision support concurrently with OLTP, without a deterioration of OLTP performance. However, if MAX_PDQPRIORITY is too low, the performance of decision-support queries can degrade.

You can set MAX_PDQPRIORITY to one of the following values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Database Server Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Turns off PDQ. DSS queries use no parallelism.</td>
<td>An integer between 0 and 100. Sets the percentage of the user-requested PDQ resources actually allocated to the query.</td>
</tr>
<tr>
<td>1</td>
<td>Fetches data from fragmented tables in parallel (parallel scans) but uses no other form of parallelism.</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>Uses all available resources for processing queries in parallel.</td>
<td></td>
</tr>
</tbody>
</table>

**MaxConnect Configuration Parameters**

Before you start IBM Informix MaxConnect, you need to specify the following configuration parameters in the `IMCCONFIG` file. This file contains both start-time and runtime parameters.

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMCLOG</td>
<td>Specifies the pathname of the MaxConnect log file.</td>
</tr>
<tr>
<td>IMCTRANSPORTS</td>
<td>Specifies the number of TCP network connections (transports) between MaxConnect and the database server.</td>
</tr>
<tr>
<td>IMCWORKERDELAY</td>
<td>Determines the time that worker threads wait to accumulate packets before they perform an aggregated send.</td>
</tr>
<tr>
<td>IMCWORKERTHREADS</td>
<td>Specifies the number of worker threads for MaxConnect.</td>
</tr>
</tbody>
</table>

MaxConnect uses the following environment variables. For more information, see the section on the configuration file in the *IBM Informix MaxConnect User’s Guide*:

- `INFORMIXDIR`
- `INFORMIXSERVER`
- `INFORMIXSQLHOSTS`
- `IMCADMIN`
- `IMCADMIN`
- `IMCCONFIG`
- `IMCSERVER`

**MIRROR**

| onconfig.std value | 0 |
The MIRROR parameter indicates whether mirroring is enabled for the database server. It is recommended that you mirror the root dbspace and the critical data as part of initialization. Otherwise, leave mirroring disabled. If you later decide to add mirroring, you can edit your configuration file to change the parameter value.

You do not have to set the MIRROR configuration parameter to the same value on both database servers in the high-availability data-replication pair. You can enable or disable mirroring on either the primary or the secondary database server independently. Do not set the MIRROR configuration parameter to 1 unless you are using mirroring.

**MIRROROFFSET**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Kilobytes</td>
</tr>
<tr>
<td>range of values</td>
<td>Any value greater than or equal to 0</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>refer to</td>
<td>Mirroring the root dbspace during initialization, in the chapter on using mirroring in the <em>IBM Informix Administrator’s Guide</em></td>
</tr>
</tbody>
</table>

In Dynamic Server, MIRROROFFSET specifies the offset into the disk partition or into the device to reach the chunk that serves as the mirror for the initial chunk of the root dbspace.

**MIRRORPATH**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>65 or fewer characters</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>
| refer to           | The following material in the *IBM Informix Administrator’s Guide*:  
  - Mirroring the root dbspace during initialization, in the chapter on using mirroring  
  - Using links, in the chapter on managing disk space |
MIRRORPATH specifies the full pathname of the mirrored chunk for the initial chunk of the root dbspace. MIRRORPATH should be a link to the chunk pathname of the actual mirrored chunk for the same reasons that ROOTPATH is specified as a link. Similarly, select a short pathname for the mirrored chunk.

**Setting Permissions (UNIX)**

You must set the permissions of the file that MIRRORPATH specifies to 660. The owner and group must both be informix.

If you use raw disk space for your mirror chunk on a UNIX platform, it is recommended that you define MIRRORPATH as a pathname that is a link to the initial chunk of the mirror dbspace, instead of entering the actual device name for the initial chunk.

---

### MSGPATH

| onconfig.std value | On UNIX: /usr/informix/online.log  
|--------------------|-----------------------------------
|                    | On Windows: online.log            |
| if not present     | On UNIX: /dev/tty                 |
| range of values    | Pathname                          |
| takes effect       | When the database server is shut down and restarted |
| utilities          | onstat -m to view the message log (For more information, see [15-122]) |
| refer to           | Message log, in the chapter on overview of database server administration in the *IBM Informix Administrator's Guide* |

MSGPATH specifies the full pathname of the message-log file. The database server writes status messages and diagnostic messages to this file during operation.

If the file that MSGPATH specifies does not exist, the database server creates the file in the specified directory. If the directory that MSGPATH specifies does not exist, the database server sends the messages to the system console.

If the file that MSGPATH specifies does exist, the database server opens it and appends messages to it as they occur.

---

### MULTIPROCESSOR

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>Platform dependent</td>
</tr>
</tbody>
</table>
| range of values    | 0 = No multiprocessor  
|                    | 1 = Multiprocessor available |
| takes effect       | When the database server is shut down and restarted |
| refer to           | CPU virtual processors, in the chapter on virtual processors in the *IBM Informix Administrator's Guide* |

Chapter 1. Configuration Parameters  1-65
If MULTIPROCESSOR is set to 0, the parameters that set processor affinity are ignored. MULTIPROCESSOR specifies whether the database server performs locking in a manner that is suitable for a single-processor computer or a multiprocessor computer.

### NETTYPE

**syntax**

NETTYPE
protocol,poll_threads,connections,VP_class
Specify the protocol as \textit{iii}ppp where:

\begin{align*}
\textit{iii}&=[\text{ipc} | \text{soc} | \text{tli}] \\
\textit{ppp}&=[\text{shm} | \text{str} | \text{tcp} | \text{spx} | \text{imc}]
\end{align*}

The \textit{protocol} value is required. You cannot use any white space in the fields, but you can omit trailing commas.

**onconfig.std values**

On UNIX: None
On Windows: onsoctcp, drsoctcp, 1, NET

**if not present**

\textit{protocol}:

On UNIX: \textit{protocol} field from the sqlhosts file (with or without the database server prefix of on, ol, or dr)
On Windows: onsoctcp

\textit{number of poll_threads}: 1
\textit{number of connections}: 50
\textit{VP_class}: NET for DSERVERALIASES;
CPU for DSERVERNAME

**separators**

Commas

**range of values**

\textit{number of poll_threads}:

On UNIX:
If \textit{VP_class} is NET, a value greater than or equal to 1
If \textit{VP_class} is CPU, 1 through \textit{num_cpu_vps}

On Windows:
Any value greater than or equal to 1

\textit{number of connections}: 1 through 32,767
\textit{VP_class}:
CPU = CPU VPs (on UNIX)
NET = Network VPs

**takes effect**

When the database server is shut down and restarted

**utilities**

\texttt{onstat -g nsc} (see \cite{[15-20]}
\texttt{onstat -g nss}
\texttt{onstat -g nta}

**refer to**

The following sections in the \textit{IBM Informix Administrator's Guide}:

- Connecting to IBM data server clients
- Allocating poll threads for an interface/protocol combination
- Setting the size of the DRDA communications buffer
Displaying DRDA thread and session information
Network protocol entry, in the chapter on client/server communications
Multiplexed connections, in the chapter on client/server communications
Network virtual processors, in the chapter on virtual processors
Should poll threads run on CPU or network virtual processors, in the chapter on virtual processors
Monitoring and tuning the number of poll threads and connections, in the IBM Informix Performance Guide

Configuring MaxConnect in IBM Informix MaxConnect User’s Guide

The NETTYPE parameter usually provides tuning options for the protocols that dbservename entries define in the sqlhosts file or registry.

Each dbservename entry in the sqlhosts file or registry is defined on either the DBSERVERNAME parameter or the DBSERVERALIASES parameter in the ONCONFIG file.

The NETTYPE configuration parameter describes a network connection as follows:
• The protocol (or type of connection)
• The number of poll threads assigned to manage the connection
• The expected number of concurrent connections
• The class of virtual processor that will run the poll threads

You can specify a NETTYPE parameter for each protocol that you want the database server to use. The following example illustrates NETTYPE parameters for two types of connections to the database server: a shared memory connection for local clients, and a network connection that uses sockets:
NETTYPE ipcshm,3,,CPU
NETTYPE soctcp,,20,NET

The NETTYPE parameter for the shared-memory connection (ipcshm) specifies three poll threads to run in CPU virtual processors. The number of connections is not specified, so it is set to 50. For ipcshm, the number of poll threads correspond to the number of memory segments.

The NETTYPE parameter for the sockets connection (soctcp) specifies that only 20 simultaneous connections are expected for this protocol and that one poll thread (because the number of poll threads is not specified) will run in a network virtual processor (in this case, NET).

Protocol
The protocol entry is the same as the nettype field in the sqlhosts file or registry, except that the database server prefix of on, ol or dr is optional. The first three characters of the protocol entry specify the interface type, and the last three characters specify the IPC mechanism or the network protocol.
Number of Poll Threads
This field specifies the number of poll threads for a specific protocol. The default value of poll_threads is 1.

If your database server has a large number of connections, you might be able to improve performance by increasing the number of poll threads. In general, each poll thread can handle approximately 200 to 250 connections.

Number of Connections
This field specifies the maximum number of connections per poll thread that can use this protocol at the same time. The default value of connections is 50. If only a few connections will be using a protocol concurrently, you might save memory by explicitly setting the estimated number of connections.

Use this formula to calculate the maximum number of connections expected. For shared memory (ipcshm), double the number of connections.
connections = max_connections / poll_threads

--- UNIX Only ---

The following example specifies 3 poll threads and 20 connections for a total of 60 shared-memory connections.
NET_TYPE ipcshm,3,20,CPU

For all net types other than ipcshm, the poll threads dynamically reallocate resources to support more connections, as needed. Avoid setting the value for the number of concurrent connections to much higher than you expect. Otherwise, you might waste system resources.

--- End of UNIX Only ---

--- Windows Only ---

On Windows, the number of connections per poll thread is used for ipcshm connections. Other protocols ignore this value. Use NET virtual processors to run the poll threads.

--- End of Windows Only ---

Class of Virtual Processor
You can set the VP_class entry to specify either CPU or NET. However, the combined number of poll threads defined with the CPU VP class for all net types cannot exceed the maximum number of CPU VPS. You should carefully distinguish between poll threads for network connections and poll threads for shared memory connections, which should run one per CPU virtual processor. TCP connections should only be in network virtual processors, and you should only have the minimum needed to maintain responsiveness. Shared memory connections should only be in CPU virtual processors and should run in every CPU virtual processor.

Note: If you use the VP classes t1i, shm, str, or soc in the settings for the VPCLASS configuration parameter, you must use the class of virtual processor class NET for the NET_TYPE configurator parameter. For more information on the VPCLASS Name on page 1-105.
For more advice on whether to run the poll threads on CPU or NET virtual processors, refer to the chapter on virtual processors in the *IBM Informix Administrator’s Guide.*

**Default Values**

It is recommended that you use NETTYPE to configure each of your connections. However, if you do not use NETTYPE, the database server uses the default values to create a single poll thread for the protocol. If the dbservername is defined by DBSERVERNAME, by default the poll thread is run by the CPU class. If the dbservername is defined by DBSERVERALIASES, the default VP class is NET.

**Multiplexed Connections**

To enable the database server to use multiplexed connections on UNIX, you must include a special NETTYPE parameter with the value sqlmux, as in the following example:

```
NETTYPE sqlmux
```

NETTYPE sqlmux does not need to be present in the ONCONFIG file. For more information on enabling multiplexed connections, see the *IBM Informix Administrator’s Guide.*

**IBM Informix MaxConnect**

If you are using IBM Informix MaxConnect, see the *IBM Informix MaxConnect User’s Guide* for how to specify the fields in the NETTYPE parameter. The onliimc and onsocimc protocols use TCP/IP to communicate with MaxConnect. You can use these protocols to either connect MaxConnect or the application clients to the database server.

### OFF_RECVRY_THREADS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>10</td>
<td>Number of recovery threads that run in parallel</td>
</tr>
<tr>
<td>units</td>
<td></td>
<td>Positive integers</td>
</tr>
<tr>
<td>range of values</td>
<td></td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>takes effect</td>
<td></td>
<td></td>
</tr>
<tr>
<td>refer to</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IBM Informix Backup and Restore Guide</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• IBM Informix Performance Guide</td>
</tr>
</tbody>
</table>

OFF_RECVRY_THREADS is the number of recovery threads used in logical recovery when the database server is offline (during a cold restore). This number of threads is also used to roll forward logical-log records in fast recovery.

Before you perform a cold restore, you can set the value of this parameter to approximately the number of tables that have a large number of transactions against them in the logical log. For single-processor computers or nodes, more than 30 to 40 threads is probably too many, because the overhead of thread management offsets the increase in parallel processing.
ON_RECVRY_THREADS

- **onconfig.std value**
  - **units**: Number of recovery threads that run in parallel
  - **range of values**: Positive integers
  - **takes effect**: When the database server is shut down and restarted

**refer to**
- IBM Informix Backup and Restore Guide
- IBM Informix Performance Guide

ON_RECVRY_THREADS is the maximum number of recovery threads that the database server uses for logical recovery when the database server is online (during a warm restore).

You can tune ON_RECVRY_THREADS to the number of tables that are likely to be recovered, because the logical-log records that are processed during recovery are assigned threads by table number. The maximum degree of parallel processing occurs when the number of recovery threads matches the number of tables being recovered.

To improve the performance of fast recovery, increase the number of fast-recovery threads with the ON_RECVRY_THREADS parameter.

---

**ON-Bar Configuration Parameters**

The following table lists the configuration parameters that apply exclusively to the ON–Bar backup and restore utility. For more information on these parameters, see the IBM Informix Backup and Restore Guide.

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAR_ACT_LOG</td>
<td>Specifies the location of the ON–Bar activity log file.</td>
</tr>
<tr>
<td>BAR_BSALIB_PATH</td>
<td>Specifies the pathname and filename of the XBSA shared library for the storage manager.</td>
</tr>
<tr>
<td>BAR_DEBUG</td>
<td>Specifies the level of debugging messages in the ON-Bar activity log.</td>
</tr>
<tr>
<td>BAR_DEBUG_LOG</td>
<td>Specifies the location of the ON-Bar debug log.</td>
</tr>
<tr>
<td>BAR_HISTORY</td>
<td>Specifies whether the sysutils database maintains a backup history.</td>
</tr>
<tr>
<td>BAR_MAX_BACKUP</td>
<td>Specifies the maximum number of parallel backup processes that are allowed for each ON-Bar command.</td>
</tr>
<tr>
<td>BAR_NB_XPORT_COUNT</td>
<td>Specifies the number of shared-memory data buffers for each backup or restore process.</td>
</tr>
<tr>
<td>BAR_PERFORMANCE</td>
<td>Specifies the level of performance statistics to print to the ON–Bar activity log.</td>
</tr>
<tr>
<td>BAR_PROGRESS_FREQ</td>
<td>Specifies in minutes how frequently the backup or restore progress messages display in the activity log.</td>
</tr>
<tr>
<td>BAR_RETRY</td>
<td>Specifies how many times ON–Bar should retry a backup or restore operation.</td>
</tr>
<tr>
<td>BAR_XFER_BUF_SIZE</td>
<td>Specifies the size in pages of the buffers.</td>
</tr>
</tbody>
</table>
### Configuration Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISM_DATA_POOL</td>
<td>Specifies the volume pool that you use for backing up storage spaces.</td>
</tr>
<tr>
<td>ISM_LOG_POOL</td>
<td>Specifies the volume pool that you use for backing up logical logs.</td>
</tr>
</tbody>
</table>

Backup and restore can also be performed using SQL API command equivalents. For more information see *IBM Informix Guide to SQL: Syntax* and *IBM Informix Administrator’s Guide*.

### ONDBSPACEDOWN

ONDBSPACEDOWN defines the action that the database server takes when any disabling event occurs on a primary chunk within a noncritical dbspace.

- **onconfig.std value**: 2
- **range of values**: 0, 1, 2
- **refer to**: Monitoring the database server for disabling I/O errors, in the chapter on consistency checking in the *IBM Informix Administrator’s Guide*

The following values are valid for this parameter:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The database server marks the dbspace as offline and continues.</td>
</tr>
<tr>
<td>1</td>
<td>The database server aborts.</td>
</tr>
</tbody>
</table>
| 2     | The database server writes the status of the chunk to the logs and waits for user input. If you set this option, but you want the database server to mark a disabled dbspace as down and continue processing, use `onmode -O` to override this ONDBSPACEDOWN setting. See "onmode -O: Override ONDBSPACEDOWN WAIT mode" on page 11-16.

### Database Server Behavior When ONDBSPACEDOWN Does Not Apply

The database server will not come online if a chunk within any critical dbspace (for example, rootdbs or logsdbs) is missing.

The value of ONDBSPACEDOWN has no effect on temporary dbspaces. For temporary dbspaces, the database server continues processing regardless of the ONDBSPACEDOWN setting. If a temporary dbspace requires fixing, you should drop and recreate it.

For a non-primary chunk within a noncritical dbspace, the behavior of the database server depends on the transaction status of the chunk when the disabling event occurs:

- **No transaction**: If no transactions are detected against that chunk, the chunk is individually marked as down. In this case, subsequent attempts to write to that chunk fail, rolling back the associated transaction. You can safely put the chunk back and then use the `onspaces -s` utility to mark the chunk as back online.
• **Transaction detected:** If there are transactions to roll forward or back, then the database server aborts with an appropriate fast recovery error. In this case, you should put the chunk back and restart the database server.

**ONLIDX_MAXMEM**

- **onconfig.std value**: 5120
- **units**: Kilobytes
- **range of values**: 16 through 4294967295
- **takes effect**: When the database server is shut down and restarted
- **utilities**: `onmode -wf`, `onmode-wm`

The ONLIDX_MAXMEM configuration parameter limits the amount of memory that is allocated to a single preimage pool and a single updater log pool. The preimage and updater log pools, `pimage_partnum` and `ulog_partnum`, are shared memory pools that are created when a CREATE INDEX ONLINE statement is executed. The pools are freed when the execution of the statement is completed.

If you specify a value for this parameter and then create a table, add rows to the table, and start to execute a CREATE INDEX ONLINE statement on a column, you can also perform other operations on the column, such as running UPDATE STATISTICS HIGH, without having memory problems.

The ONLIDX_MAXMEM configuration parameter can be changed using the `onmode -wf` option or superseded for a session with the `onmode-wm` option. For more information about `onmode`, see [“onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20](#).

**OPCACHEMAX (UNIX)**

- **onconfig.std value**: 0
- **if not present**: 128
- **units**: Kilobytes
- **range of values**: 0 through (4 * 1024 * 1024)
- **takes effect**: When the Optical Subsystem needs more memory
- **utilities**: `onstat -O` (For more information, see [15-125](#))
- **refer to**:
  - IBM Informix Optical Subsystem Guide
  - INFORMIXOPCACHE environment variable, in the IBM Informix Guide to SQL: Reference

OPCACHEMAX specifies the size of the memory cache for the Optical Subsystem. The database server stores pieces of TEXT or BYTE data in the memory cache before it delivers them to the subsystem. Use this parameter only if you use the Optical Subsystem.

The INFORMIXOPCACHE environment variable lets the client restrict the size of the optical cache that it uses.
### OPTCOMPIND

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>2</th>
</tr>
</thead>
</table>
| range of values    | 0 = When appropriate indexes exist for each ordered pair of tables, the optimizer chooses index scans (nested-loop joins), without consideration of the cost, over table scans (hash joins). This value ensures compatibility with previous versions of the database server.  
1 = The optimizer uses costs to determine an execution path if the isolation level is not Repeatable Read. Otherwise, the optimizer chooses index scans (it behaves as it does for the value 0). This setting is recommended for optimal performance.  
2 = The optimizer uses cost to determine an execution path for any isolation level. Index scans are not given preference over table scans; the optimizer bases its decision purely on cost. This value is the default if the variable is not set. |

**refer to**

- Your *IBM Informix Performance Guide*  
- OPTCOMPIND environment variable, in the *IBM Informix Guide to SQL: Reference*  
- SET ENVIRONMENT OPTCOMPIND statement to dynamically change the value of the OPTCOMPIND configuration parameter for a session, in the *IBM Informix Guide to SQL: Syntax*

OPTCOMPIND helps the optimizer choose an appropriate query plan for your application.

**Tip:** You can think of the name of the variable as arising from “OPTimizer COMPare (the cost of using) INDexes (with other methods).”

Because of the nature of *hash joins*, an application with isolation mode set to Repeatable Read might temporarily lock all records in tables that are involved in the join (even those records that fail to qualify the join) for each ordered set of tables. This situation leads to higher contention among connections. Conversely, nested-loop joins lock fewer records but provide inferior performance when the database server retrieves a large number of rows. Thus, both join methods offer advantages and disadvantages. A client application can also influence the optimizer in its choice of a join method.

### OPT_GOAL

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>0 or -1</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>refer to</td>
<td>The following manuals:</td>
</tr>
</tbody>
</table>
• ALL_ROWS and FIRST_ROWS directives and on the SET OPTIMIZATION statement, in the IBM Informix Guide to SQL: Reference
• OPT_GOAL environment variable, in the IBM Informix Guide to SQL: Syntax
• Performance issues associated with setting an optimization goal, in the IBM Informix Performance Guide

The OPT_GOAL parameter enables you to specify one of the following optimization goals for queries:
• Optimize for FIRST ROWS
• Optimize for ALL ROWS

A value of 0 sets the optimization goal to FIRST_ROWS. A value of -1 sets the optimization goal to ALL_ROWS, which is the default.

When you set the optimization goal to optimize for FIRST ROWS, you specify that you want the database server to optimize queries for perceived response time. In other words, users of interactive applications perceive response time as the time that it takes to display data on the screen. Setting the optimization goal to FIRST ROWS configures the database server to return the first rows of data that satisfy the query.

When you set the optimization goal to optimize for ALL ROWS, you specify that you want the database server to optimize for the total execution time of the query. Making ALL ROWS the optimization goal instructs the database server to process the total query as quickly as possible, regardless of how long it takes to return the first rows to the application.

You can specify the optimization goal in one of four ways:
• By query (SELECT statement)
  Use the ALL_ROWS and FIRST_ROWS directives.
• By session
  Use the SET OPTIMIZATION statement.
• By environment
  Set the OPT_GOAL environment variable.
• By database server
  Set the OPT_GOAL configuration parameter.

To determine the optimization goal, the database server examines the settings in the order shown. The first setting encountered determines the optimization goal. For example, if a query includes the ALL_ROWS directive but the OPT_GOAL configuration parameter is set to FIRST_ROWS, the database server optimizes for ALL_ROWS, as the query specifies.

---

**PC_HASHSIZE**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>Any positive non-prime number</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>
Use PC_HASHSIZE to specify the number of hash buckets in the caches that the database server uses.

PC_HASHSIZE applies to UDR cache only. For information on configuration parameters for other types of cache, see the “DS_POOLSIZE” on page 1-39 and the “DS_HASHSIZE” on page 1-36.

**PC_POOLSIZE**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>range of values</strong></td>
<td>Any positive value</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>Your IBM Informix Performance Guide</td>
</tr>
</tbody>
</table>

PC_POOLSIZE specifies the maximum number of UDRs stored in the UDR cache.

For information on configuration parameters for other types of cache, see the “DS_POOLSIZE” on page 1-39 and the “DS_HASHSIZE” on page 1-36.

**PHYSBUFF**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>units</strong></td>
<td>Kilobytes</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>4 kilobytes through ((32767 \times \text{page size} / 1024)) kilobytes.</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td><strong>utilities</strong></td>
<td><code>onstat -l buffer</code> field, first section (For more information, see 15-122)</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>Physical-log buffer, in the shared-memory chapter of the IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>

PHYSBUFF specifies the size in kilobytes of the two physical-log buffers in shared memory. Double buffering permits user threads to write to the active physical-log buffer while the other buffer is being flushed to the physical log on disk. The value of the PHYSBUFF parameter determines how frequently the database server needs to flush the physical-log buffer to the physical-log file. If RTO_SERVER_RESTART is enabled, use the 512 kilobyte default value for PHYSBUFF; if the PHYSBUFF value is less than 512 kilobytes, a warning message displays when you restart the server.

A write to the physical-log buffer is exactly one page in length. Choose a value for PHYSBUFF that is evenly divisible by the page size. If the value of PHYSBUFF is not evenly divisible by the page size, the database server rounds down the size to the nearest value that is evenly divisible by the page size.

The user-data portion of a smart large object does not pass through the physical-log buffers.
The system page size is platform-dependent on Dynamic Server. To obtain the system page size, use the commands listed in the table in “System Page Size” on page 1-22.

**PHYSDBS**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th><strong>rootdbs</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>The dbspace that ROOTNAME specifies</td>
</tr>
<tr>
<td>units</td>
<td>A dbspace</td>
</tr>
<tr>
<td>range of values</td>
<td>Up to 128 bytes. PHYSDBS must be unique, begin with a letter or underscore, and contain only letters, numbers, underscores, or $ characters.</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is initialized</td>
</tr>
<tr>
<td>refer to</td>
<td></td>
</tr>
</tbody>
</table>

PHYSDBS specifies the name of the dbspace that contains the physical log. To reduce disk contention, you can move the physical log to a dbspace other than the root dbspace.

When you initialize disk space (oninit -i), the PHYSDBS value must be equal to the ROOTDBS value.

**PHYSFILE**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th><strong>2000</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>200</td>
</tr>
<tr>
<td>units</td>
<td>Kilobytes</td>
</tr>
<tr>
<td>range of values</td>
<td>200 or more</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is initialized</td>
</tr>
<tr>
<td>onparams -p</td>
<td></td>
</tr>
<tr>
<td>refer to</td>
<td></td>
</tr>
</tbody>
</table>

PHYSFFILE specifies the size of the physical log. PHYSFFILE can be changed dynamically with the onparams utility. Restarting the server is not required for the changes to take effect.
When RTO_SERVER_RESTART is enabled, ensure that the PHYSFILE value is equal to at least 110% of the buffer pool size. A warning message prints to the message log when the PHYSFILE value is changed to less than 110% of all of the buffer pools, when the server is restarted, or when a new buffer pool is added.

### QSTATS

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
</table>
| range of values    | 0 = Disable queue statistics  
|                    | 1 = Enable queue statistics |
| takes effect       | When the database server is shut down and restarted |
| utilities          | `onstat -g qst` |
| refer to           |  
|                    | *"onstat -g qst: Print wait options for mutex and condition queues" on page 15-77* |

QSTATS specifies the ability of `onstat -g qst` to print queue statistics.

### PLOG_OVERFLOW_PATH

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td><code>$INFORMIXDIR/tmp</code></td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is brought up (shared memory is initialized)</td>
</tr>
<tr>
<td>refer to</td>
<td>Your IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>

The PLOG_OVERFLOW_PATH parameter specifies the location of the file that is used during fast recovery if the physical log file overflows. The file is `plog_extend.servernum` and by default located in `$INFORMIXDIR/tmp`. Use the full pathname to specify a different location for the file with the PLOG_OVERFLOW_PATH parameter.

### RA_PAGES

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>4 if MULTIPROCESSOR is 0; 8 if MULTIPROCESSOR is 1</td>
</tr>
<tr>
<td>units</td>
<td>Number of data pages</td>
</tr>
<tr>
<td>range of values</td>
<td>RA_THRESHOLD through BUFFERPOOL</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>refer to</td>
<td></td>
</tr>
</tbody>
</table>
|                    | *Configuring the database server to read ahead, in the shared-memory chapter of the IBM Informix Administrator’s Guide*  
|                    | *Calculating RA_PAGES and RA_THRESHOLD, in your IBM Informix Performance Guide* |
RA_PAGES specifies the number of disk pages to attempt to read ahead during sequential scans of data records. Read-ahead can greatly speed up database processing by compensating for the slowness of I/O processing relative to the speed of CPU processing.

This parameter works with the RA_THRESHOLD parameter. Specifying values that are too large can result in excessive buffer-caching activity.

### RA_THRESHOLD

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>RA_PAGES/2</td>
</tr>
<tr>
<td>units</td>
<td>Number of data pages</td>
</tr>
<tr>
<td>range of values</td>
<td>0 through (RA_PAGES - 1)</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

**refer to**

- Configuring the database server to read ahead, in the shared-memory chapter of the *IBM Informix Administrator’s Guide*
- Calculating RA_PAGES and RA_THRESHOLD, in your *IBM Informix Performance Guide*

RA_THRESHOLD is used with RA_PAGES when the database server reads during sequential scans of data records. RA_THRESHOLD specifies the read-ahead threshold; that is, the number of unprocessed data pages in memory that signals the database server to perform the next read-ahead.

If the value of RA_THRESHOLD is greater than the value of RA_PAGES, RA_THRESHOLD has a value of RA_PAGES/2.

Specifying values that are too large for RA_PAGES and RA_THRESHOLD can result in excessive buffer-caching activity.

### RESIDENT

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>-1 to 99</td>
</tr>
<tr>
<td>if not present</td>
<td>0</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

Certain platforms have different values. For information, see your machine notes.
utilities

**onmode -r** (see "onmode -n, -r: Change shared-memory residency" on page 11-15)

refer to

The following topics in the *IBM Informix Administrator's Guide* for a discussion of residency:

- Resident portion of shared memory, in the shared-memory chapter
- Setting database server shared-memory configuration parameters, in the chapter on managing shared memory

The RESIDENT parameter specifies whether resident and virtual segments of shared memory remain resident in operating-system physical memory.

Some systems allow you to specify that the resident portion of shared memory must stay (be resident) in memory at all times. If your operating system supports forced residency, you can specify that resident and virtual segments of shared memory not be swapped to disk.

**Warning:** Before you decide to enforce residency, verify that the amount of physical memory available is sufficient to execute all required operating-system and application processes. If insufficient memory is available, a system hang could result that requires a reboot.

---

**RESTARTABLE_RESTORE**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th><strong>ON</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>if not present</strong></td>
<td><strong>ON</strong></td>
</tr>
</tbody>
</table>
| **range of values**    | OFF = restartable restore is disabled  
|                        | ON = restartable restore is enabled |
| **takes effect**       | When the database server is shut down and restarted |
| **refer to**           | *IBM Informix Backup and Restore Guide* |

If you set RESTARTABLE_RESTORE to ON, you enable the database server to restart a failed physical or cold logical restore at the point at which the failure occurred. To perform a restartable restore with ON–Bar, use the **onbar -RESTART** command.

Increase the size of your physical log if you plan to use restartable restore. For more information, see "PHYSFILE" on page 1-76. Although a restartable restore slows down the logical restore if many logs need to be restored, you save a lot of time from not having to repeat the entire restore.

**Important:** If the database server fails during a warm logical restore, you must repeat the entire restore. If the database server is still running, use **onbar -r -I** to complete the restore.

If you do a cold restore on systems that are not identical, you can assign new pathnames to chunks, and you can rename devices for critical chunks during the restore. You must perform a level-0 archive after the rename and restore operation completes. For details, see the *IBM Informix Backup and Restore Guide*
The database server uses physical recovery and logical recovery to restore data as follows:

- **Physical recovery.** The database server writes data pages from the backup media to disk. This action leaves the storage spaces consistent to the point at which it was originally backed up. However, the backup times for each storage space are usually different. A restartable restore is restartable to the level of a storage space. If only some chunks of a storage space are restored when the restore fails, the entire storage space needs to be recovered again when you restart the restore.

- **Logical recovery.** The database server replays logical-log records on media to bring all the storage spaces up to date. At the end of logical recovery, all storage spaces are consistent to the same point.

### ROOTNAME

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>onconfig.std</strong></td>
<td><em>value</em></td>
</tr>
<tr>
<td><strong>rootdbs</strong></td>
<td>A dbspace</td>
</tr>
<tr>
<td><strong>units</strong></td>
<td>Up to 128 bytes. ROOTNAME must begin with a letter or underscore and must contain only letters, numbers, underscores, or $ characters.</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>When disk is initialized (destroys all data)</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>Allocating disk space, in the chapter on managing disk space in the IBM Informix Administrator’s Guide</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>Allocating disk space, in the chapter on managing disk space in the IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>

ROOTNAME specifies a name for the root dbspace for this database server configuration.

The name must be unique among all dbspaces that the database server manages. It is recommended that you select a name that is easily recognizable as the root dbspace.

### ROOTOFFSET

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>onconfig.std</strong></td>
<td><em>value</em></td>
</tr>
<tr>
<td><strong>units</strong></td>
<td>Kilobytes</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>Any value greater than or equal to 0</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When disk is initialized (destroys all data)</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>Allocating raw disk space on UNIX, in the chapter on managing disk space in the IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>

ROOTOFFSET specifies the offset into an allocation of disk space (file, disk partition, or device) at which the initial chunk of the root dbspace begins.

**UNIX Only**

On some UNIX platforms, it is not valid to set ROOTOFFSET to 0. When this parameter is set incorrectly, you must reinitialize disk space and reload data to resume proper operation of the database server. Before you configure the database server, you must review the applicable information on managing disk space in the IBM Informix Administrator’s Guide.
server, always check your machine notes file for information about proper settings.

End of UNIX Only

### ROOTPATH

| **onconfig.std value** | On UNIX: `/dev/online_root`  
| **range of values** | Pathname  
| **takes effect** | When disk is initialized (destroys all data)  
| **refer to** | The following material in the chapter on managing disk space in the *IBM Informix Administrator’s Guide*  
| **** | • Allocating disk space  
| **** | • Creating links for raw devices

ROOTPATH specifies the full pathname, including the device or filename, of the initial chunk of the root dbspace. ROOTPATH is stored in the reserved pages as a chunk name.

On UNIX, you must set the permissions of the file that ROOTPATH specifies to 660, and the owner and group must both be `informix`. On Windows, a member of the `Informix-Admin` group must own the file that ROOTPATH specifies.

---

**UNIX Only**

If you use unbuffered disk space for your initial chunk on UNIX, it is recommended that you define ROOTPATH as a pathname that is a link to the initial chunk of the root dbspace instead of entering the actual device name for the initial chunk.

---

End of UNIX Only

### ROOTSIZE

| **onconfig.std value** | UNIX: 100,000 Windows: 100,000  
| **if not present** | 0  
| **units** | Kilobytes  
| **range of values** | 50,000 through maximum capacity of the storage device  
| **takes effect** | When disk is initialized (destroys all data)  
| **refer to** | Calculating the size of the root dbspace, in the chapter on where is data stored in the *IBM Informix Administrator’s Guide*

ROOTSIZE specifies the size of the initial chunk of the root dbspace, expressed in kilobytes. The size that you select depends on your immediate plans for your database server.

To change ROOTSIZE after you initialize the database server, completely unload and reload your data.
**RTO_SERVER_RESTART**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>0 (disabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>units</strong></td>
<td>seconds</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>0 = disabled</td>
</tr>
<tr>
<td></td>
<td>60 to 1800</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When the database server is stopped and restarted. When the RTO_SERVER_RESTART value is changed by using onmode -wm or onmode -wf.</td>
</tr>
<tr>
<td><strong>utilities</strong></td>
<td>oncheck -pr</td>
</tr>
<tr>
<td></td>
<td>onmode -wf or onmode -wm</td>
</tr>
<tr>
<td></td>
<td>onparams</td>
</tr>
</tbody>
</table>

**refer to**

- “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20
- Non-blocking checkpoints information in IBM Informix Administrator’s Guide
- Effects of configuration on I/O activity in IBM Informix Performance Guide

RTO_SERVER_RESTART enables you to use recovery time objective (RTO) standards to set the amount of time, in seconds, that Dynamic Server has to recover from a problem after you restart Dynamic Server and bring the server into online or quiescent mode.

**SBSPACENAME**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>if not present</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>Up to 128 bytes. SBSPACENAME must be unique, begin with a letter or underscore, and contain only letters, digits, underscores, or $ characters.</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When shared memory is reinitialized</td>
</tr>
<tr>
<td><strong>utilities</strong></td>
<td>onspaces -c -S</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>Using onspaces to “onspaces -c -S: Create an sbspace” on page 14-11</td>
</tr>
<tr>
<td></td>
<td>“SBSPACETEMP” on page 1-83</td>
</tr>
<tr>
<td></td>
<td>“SYSSBSPACENAME” on page 1-96</td>
</tr>
<tr>
<td></td>
<td>“Sbspace Structure” on page 4-23</td>
</tr>
<tr>
<td></td>
<td>What is an sbspace, in the chapter on data storage in the IBM Informix Administrator’s Guide</td>
</tr>
<tr>
<td></td>
<td>Altering sbspace characteristics, in the chapter on managing data on disk in the IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>
Assigning a smart large object to an subspace, in the section on the CREATE TABLE and ALTER TABLE statements, in the *IBM Informix Guide to SQL: Syntax*

Creating an subspace for Enterprise Replication usage, in the *IBM Informix Dynamic Server Enterprise Replication Guide*

Using multirepresentational data, in the *IBM Informix DataBlade API Programmer’s Guide*

**SBSPACENAME** specifies the name of the default subspace. If your database tables include smart-large-object columns that do not explicitly specify a storage space, that data is stored in the subspace that SBSPACENAME specifies.

You must create the default subspace with the **onspaces -c -S** utility before you can use it. The database server validates the name of the default subspace when one of the following occurs:

- You specify the default subspace as the storage option for a CLOB or BLOB column in the PUT clause of the CREATE TABLE or ALTER TABLE statement.
- The database server attempts to write a smart large object to the default subspace when no subspace was specified for the column.
- You store multirepresentational data in the default subspace.

**JAVA Language Support**

If you are using IBM Informix Dynamic Server with J/Foundation, you must provide a smart large object where the database server can store the Java archive (JAR) files. These JAR files contain your Java user-defined routines (UDRs). It is suggested that when you use Java UDRs, you create separate subspace for storing smart large objects.

**Warning:** When you use Enterprise Replication, you must set the **CDR_QDATA_SBSPACE** parameter and create the subspace before you define the replication server.

**SBSPACETEMP**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>Temporary smart large objects are stored in a standard subspace.</td>
</tr>
<tr>
<td>range of values</td>
<td>Up to 128 bytes. SBSPACETEMP must be unique, begin with a letter or underscore, and contain only letters, digits, underscores, or $ characters.</td>
</tr>
<tr>
<td>takes effect</td>
<td>When shared memory is reinitialized</td>
</tr>
<tr>
<td>utilities</td>
<td>onspaces</td>
</tr>
<tr>
<td>refer to</td>
<td>&quot;onspaces -c -S: Create an subspace” on page 14-11</td>
</tr>
<tr>
<td></td>
<td>&quot;SBSPACENAME” on page 1-82</td>
</tr>
</tbody>
</table>

Chapter 1. Configuration Parameters 1-83
- Temporary sbspaces, in the chapter on data storage in the IBM Informix Administrator’s Guide
- Creating a temporary sbspace, in the chapter on managing disk space in the IBM Informix Administrator’s Guide
- Using temporary smart large objects, in the IBM Informix DataBlade API Programmer’s Guide

SBSPACETEMP specifies the name of the default temporary sbspace for storing temporary smart large objects without metadata or user-data logging. If you store temporary smart large objects in a standard sbspace, the metadata is logged.

### SDS_ENABLE

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Refer to</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>0</td>
<td>Configuration parameters in the IBM Informix Dynamic Server Administrator’s Guide.</td>
</tr>
<tr>
<td>if not present</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>range of values</td>
<td>0, 1 0 = disable SD secondary server functionality, 1 = enable SD secondary server functionality</td>
<td></td>
</tr>
<tr>
<td>takes effect</td>
<td>when database server is shut down and restarted or when onmode -wf SDS_ENABLE=&lt;value&gt; is run.</td>
<td></td>
</tr>
</tbody>
</table>

SDS_ENABLE enables SD secondary server functionality. You must set SDS_ENABLE to 1 (enable) on both the primary server and the SD secondary server to enable SD secondary server functionality.

### SDS_PAGING

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Refer to</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>path1,path2</td>
<td>Configuration parameters in the IBM Informix Dynamic Server Administrator’s Guide.</td>
</tr>
<tr>
<td>if not present</td>
<td></td>
<td></td>
</tr>
<tr>
<td>range of values</td>
<td>path and filename of two paging files separated by a comma.</td>
<td></td>
</tr>
<tr>
<td>takes effect</td>
<td>when SD secondary server is started</td>
<td></td>
</tr>
</tbody>
</table>

SDS_PAGING specifies the location of two files that will act as buffer paging files. If SDS_PAGING is not set, the SD secondary server may fail to start.

### SDS_TEMPDBS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Refer to</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>dbspaceName,path,pagesize,offset,size</td>
<td></td>
</tr>
<tr>
<td>if not present</td>
<td>Error message displayed: &quot;Can not initialize SDS - Missing or invalid entry for SDS_TEMPDBS.&quot; SD secondary server will not start.</td>
<td></td>
</tr>
</tbody>
</table>
units            pagesize, offset, and size specified in kB. All temporary SDS dbspaces must be the same page size.
range of values  0 on the primary database server, 1 to 16 dbspace entries on an SD secondary server
takes effect    When the SD secondary server is started
refer to
                • Configuration parameters in the IBM Informix Dynamic Server Administrator’s Guide.

SDS_TEMPDBS specifies information that the SD secondary server uses to dynamically create temporary dbspaces. These temporary dbspaces are created (or initialized if the dbspaces existed previously), when the SD secondary server starts. The temporary dbspaces are used for creating temporary tables. There must be at least one occurrence of SDS_TEMPDBS in the ONCONFIG file of the SD secondary server in order for the SD secondary server to start.

Up to 16 SD secondary dbspaces may be specified in the ONCONFIG file by using multiple occurrences of SDS_TEMPDBS. For each occurrence of SDS_TEMPDBS in the ONCONFIG file, dspaceName must be unique and the combination of path, offset, and size must not cause any overlap with existing chunks or among SDS_TEMPDBS spaces. In addition, dspaceName must be unique among all existing dbspaces, blobspaces, and sblobspaces, including those (possibly disabled) temp spaces inherited from a primary server.

SDS_TEMPDBS is not used on the primary server.

---

**SDS_TIMEOUT**

onconfig.std value 20
if not present 10
range of values 2 to 2147483647
units seconds
takes effect when shared disk functionality is enabled on the primary server
refer to
                • The onmode -wf command
                • Configuration parameters in the IBM Informix Dynamic Server Administrator’s Guide.

SDS_TIMEOUT specifies the amount of time in seconds that the primary server will wait for a log position acknowledgement to be sent from the SD secondary server. If there is no log position acknowledgement received from the SD secondary server in the specified amount of time, then the primary server will be disconnected from the SD secondary server and continue. After waiting for the SDS_TIMEOUT number of seconds, the primary server will start removing SD secondary servers if page flushing has timed out while waiting for an SD secondary server.
### SECURITY_LOCALCONNECTION

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Value</th>
<th>Range of Values</th>
<th>Takes Effect</th>
<th>Refer to</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td><code>value</code></td>
<td>1</td>
<td>0, 1, 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = no security checking occurs</td>
<td>When the database server is shut down and restarted</td>
<td>Role of the SERVERNUM configuration parameter, in the multiple-residency chapter of the IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>

SECURITY_LOCALCONNECTION lets you verify security on local connections by verifying that the ID of the local user who is running a program is the same ID of the user who is trying to access the database.

### SERVERNUM

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Value</th>
<th>Range of Values</th>
<th>Takes Effect</th>
<th>Refer to</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td><code>value</code></td>
<td>0</td>
<td>0 through 255</td>
<td>Role of the SERVERNUM configuration parameter, in the multiple-residency chapter of the IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>

SERVERNUM specifies a relative location in shared memory. The value that you choose must be unique for each database server on your local computer. The value does not need to be unique on your network. Because the value 0 is included in the onconfig.std file, it is suggested that you choose a value other than 0 to avoid inadvertent duplication of SERVERNUM.

### SHMADD

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Value</th>
<th>Range of Values</th>
<th>Units</th>
<th>Takes Effect</th>
<th>Utilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td><code>value</code></td>
<td>8192</td>
<td>Kilobytes</td>
<td>When the database server is shut down and restarted</td>
<td><code>onstat -g seg</code> (Use the onstat -g seg command to display the number of shared-memory segments)</td>
</tr>
</tbody>
</table>
that the database server is currently using. For more information, see [15-24]

The following material in the IBM Informix Administrator’s Guide:
- Virtual portion of shared memory, in the shared-memory chapter
- Monitoring shared-memory segments with `onstat -g seg`, in the managing memory chapter

SHMADD specifies the size of a segment that is dynamically added to the virtual portion of shared memory.

It is more efficient to add memory in large segments, but wasteful if the added memory is not used. Also, the operating system might require you to add memory in a few large segments rather than many small segments.

The following table contains recommendations for setting the initial value of SHMADD.

<table>
<thead>
<tr>
<th>Amount of Physical Memory</th>
<th>Recommended SHMADD Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Less than 256 megabytes</td>
<td>8192</td>
</tr>
<tr>
<td>Between 256 megabytes and 512 megabytes</td>
<td>16,384</td>
</tr>
<tr>
<td>Greater than 512 megabytes</td>
<td>32,768</td>
</tr>
</tbody>
</table>

---

### SHMBASE

**onconfig.std value**

On UNIX: Platform dependent
On Windows: 0xC0000000L

*units*

Address

*range of values*

Positive integers

*takes effect*

When the database server is shut down and restarted

*utilities*

To see the shared-memory segment addresses, use the `onstat -g seg` command.

*refer to*

Setting operating-system shared-memory configuration parameters, in the chapter on managing shared memory in the IBM Informix Administrator’s Guide

SHMBASE specifies the base address where shared memory is attached to the memory space of a virtual processor. The addresses of the shared-memory segments start at the SHMBASE value and grow until the upper-bound limit, which is platform specific.

Do not change the value of SHMBASE. The `onconfig.std` value for SHMBASE depends on the platform and whether the processor is 32-bit or 64-bit. For information on which SHMBASE value to use, see the machine notes.
**SHMTOTAL**

- **onconfig.std value**: 0
- **units**: Kilobytes
- **range of values**: Integer greater than or equal to 1
- **takes effect**: When the database server is shut down and restarted
- **refer to**: How much shared memory the database server needs, in the shared-memory chapter of the *IBM Informix Administrator’s Guide*

SHMTOTAL specifies the total amount of shared memory (resident, virtual, communications, and virtual extension portions) to be used by the database server for all memory allocations. The `onconfig.std` value of 0 implies that no limit on memory allocation is stipulated.

SHMTOTAL enables you to limit the demand for memory that the database server can place on your system. However, applications might fail if the database server requires more memory than the limit imposed by SHMTOTAL. When this situation occurs, the database server writes the following message in the message log:

`size of resident + virtual segments xx + yy > zz total allowed by configuration parameter SHMTOTAL`

This message includes the following values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xx</td>
<td>Current size of resident segments</td>
</tr>
<tr>
<td>yy</td>
<td>Current size of virtual segments</td>
</tr>
<tr>
<td>zz</td>
<td>Total shared memory required</td>
</tr>
</tbody>
</table>

---

**UNIX Only**

Set the operating-system parameters for maximum shared-memory segment size, typically SHMMAX, SHMSIZE, or SHMALL, to the total size that your database server configuration requires. For information on the amount of shared memory that your operating system allows, see the machine notes.

---

**SHMVIRT_ALLOCSEG**

- **onconfig.std value**: 0
- **units**: First parameter: percentage of memory used OR whole number of kilobytes remaining in the server
  Second parameter: alarm level
- **range of values**: First parameter:
  Any percentage from .40 to .99 OR a whole number of kilobytes from 256 to 1,000,000
  Second parameter:
  Alarm level values from 1 to 5 where:
  1 = Not noteworthy
<table>
<thead>
<tr>
<th>Dimension</th>
<th>Information</th>
<th>Attention</th>
<th>Emergency</th>
<th>Fatal</th>
</tr>
</thead>
</table>

When the database server is shut down and restarted, a new memory segment is allocated and the alarm level is activated if the server cannot allocate the new memory segment, thus ensuring that the server never runs out of memory. Once the alarm level is activated, it will repeat every thirty minutes if a new memory segment cannot be allocated.

**SHMVIRTSIZE**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>8000 on UNIX and 8192 on Windows</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>If SHMADD is present: SHMADD If SHMADD is not present: 8</td>
</tr>
<tr>
<td>units</td>
<td>Kilobytes</td>
</tr>
<tr>
<td>range of values</td>
<td>32-bit platforms:</td>
</tr>
<tr>
<td></td>
<td>Positive integer with a maximum value of 2 gigabytes</td>
</tr>
<tr>
<td></td>
<td>64-bit platforms:</td>
</tr>
<tr>
<td></td>
<td>Positive integer with a maximum value of 4 terabytes</td>
</tr>
<tr>
<td></td>
<td>The maximum value might be less on some platforms due to operating-system limitations. For the actual maximum value for your UNIX platform, see the machine notes.</td>
</tr>
</tbody>
</table>

When the database server is shut down and restarted, a new memory segment is allocated.

Utilities

- `onstat -g seg` (see Chapter 1-24)

Refer to

- Virtual portion of shared memory, in the shared-memory chapter of the IBM Informix Administrator’s Guide
- Chapter on configuration effects on memory utilization, in your IBM Informix Performance Guide

SHMVIRTSIZE specifies the initial size of a virtual shared-memory segment. Use the following algorithm to determine the size of the virtual portion of shared memory:

\[
\text{shmvirtsize} = \text{fixed overhead} + \text{shared structures} + \text{mncs} * \text{private structures} + \text{other buffers}
\]
This algorithm includes the following values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fixed_overhead</td>
<td>Global pool + thread pool after booting (partially dependent on the number of virtual processors)</td>
</tr>
<tr>
<td>shared_structures</td>
<td>AIO vectors + sort memory + dbspace backup buffers + dictionary size + size of stored-procedure cache + histogram pool + other pools (See the onstat display.)</td>
</tr>
<tr>
<td>mncs</td>
<td>Maximum number of concurrent sessions</td>
</tr>
<tr>
<td>private_structures</td>
<td>Stack (generally 32 kilobytes but dependent on recursion in SPL routines and triggers) + heap (about 30 kilobytes) + session-control-block structures</td>
</tr>
</tbody>
</table>

If messages in the message file indicate that the database server is adding segments to the virtual portion of shared memory for you, add the amount that these messages indicate to the value of SHMVIRTSIZE. It is recommended that you initially create a virtual portion of shared memory of a size that is more than sufficient for your daily processing, if possible.

Use the onstat -g seg command to determine peak usage and lower the value of SHMVIRTSIZE accordingly.

**SINGLE_CPU_VP**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>0 = running with multiple CPU VP; Any nonzero value = running with one CPU VP</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>refer to</td>
<td>Running on a single-processor computer, in the chapter on virtual processors in the IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>

SINGLE_CPU_VP specifies whether or not the database server is running with only one CPU virtual processor.

Setting SINGLE_CPU_VP to nonzero allows the database server to use optimized code based on the knowledge that only one CPU virtual processor is running. It enables the database server to bypass many of the mutex calls that it must use when it runs multiple CPU virtual processors.

It is strongly recommended that you set this parameter when the database server will run only one CPU virtual processor. Depending on the application and workload, setting this parameter can improve performance by up to 10 percent.

If you set SINGLE_CPU_VP to nonzero and try to add a CPU virtual processor, you receive one of the following messages:

- onmode: failed when trying to change the number of classname VP(s) by n.
- onmode: failed when trying to change the number of cpu virtual processors by n.
If you set SINGLE_CPU_VP to nonzero and then attempt to bring up the database server with VPCLASS cpu,num set to a value greater than 1, you receive the following error message, and the database server initialization fails:

Cannot have SINGLE_CPU_VP non-zero and CPU VPs greater than 1.

User-Defined VP Classes and SINGLE_CPU_VP

Important: Dynamic Server treats user-defined virtual-processor classes as if they were CPU virtual processors. Thus, if you set SINGLE_CPU_VP to nonzero, you cannot create any user-defined virtual-processor classes.

If you set this parameter to nonzero and then attempt to bring up the database server with the VPCLASS cpu value for num set to a value greater than 1, you receive the following error message, and the database server initialization fails:

Cannot have SINGLE_CPU_VP non-zero and CPU VPs greater than 1.

If you set this parameter to nonzero and then attempt to bring up the database server with a user-defined VPCLASS, you receive the following error message, and the database server initialization fails:

oninit: Cannot have SINGLE_CPU_VP non-zero and user-defined VP classes

SQLTRACE

Syntax

SQLTRACE [Level=off|low|med|high],
[Ntraces=number of traces],
[Size=size of each trace buffer],[Mode=global|user]

Onconfig.std Value

OFF

Range of Values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level</td>
<td>Amount of information traced. The default is off.</td>
</tr>
<tr>
<td>Ntraces</td>
<td>Number of SQL statements to trace before reusing the resources. The range is 500 to 2147483647.</td>
</tr>
<tr>
<td>Size</td>
<td>Maximum size of variable length data to be stored. The range is 1 Kilobyte to 100 Kilobytes.</td>
</tr>
<tr>
<td>Mode</td>
<td>Type of tracing performed. Global (default) = all users; user = specific user.</td>
</tr>
</tbody>
</table>

Takes Effect

When the database server is shut down and restarted

Utilities

onstat -g his

Refer To

Specifying SQL Tracing Information by Using the SQLTRACE Configuration Parameter information in the IBM Informix Administrator’s Guide

The SQLTRACE parameter controls the startup environment of the SQL Trace facility.
STACKSIZE

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std value</td>
<td>32 for 32-bit database servers</td>
</tr>
<tr>
<td></td>
<td>64 for 64-bit database servers</td>
</tr>
<tr>
<td>units</td>
<td>Kilobytes</td>
</tr>
<tr>
<td>range of values</td>
<td>32 through limit determined by</td>
</tr>
<tr>
<td></td>
<td>the database server configuration and the amount of memory available</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

The STACKSIZE parameter specifies the stack size for the database server user threads. The value of STACKSIZE does not have an upper limit, but setting a value that is too large wastes virtual memory space and can cause swap-space problems.

For 32-bit platforms, the default STACKSIZE value of 32 kilobytes is sufficient for nonrecursive database activity. For 64-bit platforms, the recommended STACKSIZE value is 64 kilobytes. When the database server performs recursive database tasks, as in some SPL routines, for example, it checks for the possibility of stack-size overflow and automatically expands the stack.

User threads execute user-defined routines. To increase the stack size for a particular routine, use the stack modifier on the CREATE FUNCTION statement.

**Warning:** Setting the value of STACKSIZE too low can cause stack overflow, the result of which is undefined but usually undesirable.

STAGEBLOB

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std value</td>
<td>None</td>
</tr>
<tr>
<td>range of values</td>
<td>Up to 128 bytes. STAGEBLOB must be unique, begin with a letter or underscore, and contain only digits, letters, underscores, or $ characters.</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

Use this parameter only if you are storing TEXT or BYTE data on optical storage with the Optical Subsystem. This parameter has no effect on ordinary blobspaces or sbspaces.

STAGEBLOB is the blobspace name for the area where the Optical Subsystem stages TEXT and BYTE data that is destined for storage on optical disk.

STMT_CACHE

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std value</td>
<td>None (this parameter is not in onconfig.std)</td>
</tr>
</tbody>
</table>
if not present  0
range of values  0, 1, or 2
takes effect  When the database server is shut down and restarted
utilities  onmode -e
refer to
  • “onmode -e: Change usage of the SQL statement cache” on page 11-11
  • Improving query performance, in the IBM Informix Performance Guide

STMT_CACHE determines whether the database server uses the SQL statement cache. You can enable the SQL statement cache in one of two modes:
• Always use the SQL statement cache unless a user explicitly specifies not to use it. Set the STMT_CACHE configuration parameter to 2 or onmode -e ON.
• Use the SQL statement cache only when a user explicitly specifies to use it. Set the STMT_CACHE configuration parameter to 1 or onmode -e ENABLE.

The following table describes the possible values.

<table>
<thead>
<tr>
<th>Possible Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SQL statement cache not used (equivalent to onmode -e OFF).</td>
</tr>
<tr>
<td>1</td>
<td>SQL statement cache enabled, but user sessions do not use the cache. Users use the cache only if they set the environment variable STMT_CACHE to 1 or execute the SQL statement SET STATEMENT CACHE ON.</td>
</tr>
<tr>
<td>2</td>
<td>SQL statement cache turned on. All statements are cached. To turn off statement caching, set the environment variable STMT_CACHE to 0 or execute the SQL statement SET STATEMENT CACHE OFF.</td>
</tr>
</tbody>
</table>

**STMT_CACHE_HITS**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>None (this parameter is not in onconfig.std)</td>
<td></td>
</tr>
</tbody>
</table>
| if not present  0
units  Integer
range of values  Any value greater than or equal to 0
takes effect  When the database server is shut down and restarted
utilities  onmode -W STMT_CACHE_HITS
onstat (See “onstat -g ssc: Print SQL statement occurrences” on page 15-104)
refer to
  • “onmode -W: Change settings for the SQL statement cache” on page 11-20
Improving query performance, in the IBM Informix Performance Guide

STMT_CACHE_HITS specifies the number of hits (references) to a statement before it is fully inserted in the SQL statement cache. The following table describes the possible values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Fully insert all qualified statements in the SQL statement cache.</td>
</tr>
<tr>
<td>&gt;0</td>
<td>The first time a user issues a unique statement, the database server inserts a key-only entry in the cache that identifies the statement. Subsequent identical statements increment the hit count of the key-only cache entry. When the hit count of the key-only cache entry reaches the specified number of hits, the database server fully inserts the statement in the cache. Set hits to 1 or more to exclude ad hoc queries from entering the cache.</td>
</tr>
</tbody>
</table>

STMT_CACHE_NOLIMIT

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>1</td>
</tr>
<tr>
<td>range of values</td>
<td>0 or 1</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

Utilities

onmode -W STMT_CACHE_NOLIMIT

onstat (See onstat -g ssc: Print SQL statement occurrences on page 15-104)

Refer to

- onmode -W: Change settings for the SQL statement cache on page 11-20
- Improving query performance, in the IBM Informix Performance Guide

STMT_CACHE_NOLIMIT controls whether to insert qualified statements into the SQL statement cache. The following table describes the possible values.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Prevents statements from being inserted in the cache. The cache can grow beyond the size limit if most of the statements in the cache are currently in use, because the cache cleaning cannot catch up with the insert rate. If you are concerned about memory usage, turn off STMT_CACHE_NOLIMIT to prevent the database server from allocating a large amount of memory for the cache.</td>
</tr>
<tr>
<td>1</td>
<td>Always insert statements in the SQL statement cache regardless of the cache size.</td>
</tr>
</tbody>
</table>
STMT_CACHE_NUMPOOL

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None (this parameter is not in onconfig.std)</th>
</tr>
</thead>
<tbody>
<tr>
<td>if not present</td>
<td>1</td>
</tr>
<tr>
<td>units</td>
<td>Positive integer</td>
</tr>
<tr>
<td>range of values</td>
<td>1 to 256</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>refer to</td>
<td>Improving query performance, in the IBM Informix Performance Guide</td>
</tr>
</tbody>
</table>

STMT_CACHE_NUMPOOL specifies the number of memory pools for the SQL statement cache. To obtain information about these memory pools, use `onstat -g ssc pool`.

Because the database server does not insert not all statements that allocate memory from the memory pools in the cache, the cache size might be smaller than the total size of the memory pools.

STMT_CACHE_SIZE

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None (this parameter is not in onconfig.std)</th>
</tr>
</thead>
<tbody>
<tr>
<td>default size of SQL statement cache</td>
<td>512 kilobytes (524288 bytes)</td>
</tr>
<tr>
<td>units</td>
<td>Kilobytes</td>
</tr>
<tr>
<td>range of values</td>
<td>Positive integer</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>utilities</td>
<td><code>onmode -W STMT_CACHE_SIZE</code></td>
</tr>
<tr>
<td></td>
<td><code>onstat -g ssc (Maxsize field)</code></td>
</tr>
<tr>
<td>refer to</td>
<td>improving query performance, in the IBM Informix Performance Guide</td>
</tr>
</tbody>
</table>

The STMT_CACHE_SIZE configuration parameter specifies the size of the SQL statement caches in kilobytes. The new cache size takes effect the next time a statement is added to a cache.

SYSALARMPROGRAM

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>On UNIX: /usr/informix/etc/evidence.sh On Windows: %INFORMIXDIR%\etc\evidence.bat</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>Pathname</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>
Set SYSALARMPROGRAM to the full pathname of the evidence.sh script. The database server executes evidence.sh when a database server failure occurs. Technical Support uses the output from the evidence.sh script to diagnose the cause of a database server failure.

On Windows, you must enable command extensions for evidence.bat to successfully complete. You can enable and disable the extensions for the Command Prompt you are working in by issuing the following commands:

- **Enable**: cmd /x
- **Disable**: cmd /y

You can also enable and disable command extensions from the Windows XP registry:

**Table 1-1. Enabling command extensions from the Windows registry**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hive</td>
<td>HKEY_CURRENT_USER</td>
</tr>
<tr>
<td>Key</td>
<td>Software\Microsoft\Command Processor</td>
</tr>
<tr>
<td>Name</td>
<td>EnableExtensions</td>
</tr>
<tr>
<td>Type</td>
<td>REG_DWORD</td>
</tr>
<tr>
<td>Values</td>
<td>0 (disable), 1 (enable)</td>
</tr>
</tbody>
</table>

**SYSSBSPACENAME**

- **onconfig.std value**: None
  
- **if not present**: 0
  
- **range of values**: Up to 128 bytes. SYSSBSPACENAME must be unique, begin with a letter or underscore, and contain only digits, letters, underscores, or $ characters.

- **takes effect**: When disk is initialized (destroys all data)

- **utilities**
  
- **onspaces**
  
- **refer to**
  
  - “onspaces -c -S: Create an sbspace” on page 14-11
  - “Sbspace Structure” on page 4-23
  - Updating statistics, in the chapter on individual query performance in your IBM Informix Performance Guide
  - Sbspace characteristics, in the chapter on configuration effects on I/O in your IBM Informix Performance Guide
  - Writing user-defined statistics, in the performance chapter in IBM Informix User-Defined Routines and Data Types Developer’s Guide


Providing statistics data for a column, in the IBM Informix DataBlade API Programmer’s Guide

“SBSPACENAME” on page 1-82 (specifies the name of the default sbspace)

SYSSBSPACENAME specifies the name of the sbspace in which the database server stores statistics that the UPDATE STATISTICS statement collects for certain user-defined data types. Normally, the database server stores statistics in the sysdistrib system catalog table.

Because the data distributions for user-defined data types can be large, you have the option to store them in an sbspace instead of in the sysdistrib system catalog table. If you store the data distributions in an sbspace, use DataBlade API or ESQL/C functions to examine the statistics.

Even though you specify an sbspace with the SYSSBSPACENAME parameter, you must create the sbspace with the -c -s option of the onspaces utility before you can use it. The database server validates the name of this sbspace when one of the following occurs:

- The database server attempts to write data distributions of the multirepresentational type to SYSSBSPACENAME when it executes the UPDATE STATISTICS statement with the MEDIUM or HIGH keywords.
- The database server attempts to delete data distributions of the multirepresentational type to SYSSBSPACENAME when it executes the UPDATE STATISTICS statement with the DROP DISTRIBUTIONS keywords.

Although you can store smart large objects in the sbspace specified in SYSSBSPACENAME, keeping the distribution statistics and smart large objects in separate sbspaces is recommended because:

- You avoid disk contention when queries are accessing smart large objects and the optimizer is using the distributions to determine a query plan.
- Disk space takes longer to fill up when each sbspace is used for a different purpose.

### TAPEBLK

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Kilobytes</td>
</tr>
<tr>
<td>range of values</td>
<td>Values greater than pagesize/1024</td>
</tr>
<tr>
<td>takes effect</td>
<td>For ontape: when you execute ontape For onload and onunload: when the database server is shut down and restarted</td>
</tr>
<tr>
<td>refer to</td>
<td>Using onload and onunload, in the IBM Informix Migration Guide Using ontape, in the IBM Informix Backup and Restore Guide “LTAPEBLK” on page 1-58</td>
</tr>
</tbody>
</table>
TAPEBLK specifies the block size of the device to which ontape writes during a storage-space backup. TAPEBLK also specifies the default block size of the device to which data is loaded or unloaded when you use the onload or onunload utilities. If you are using onload or onunload, you can specify a different block size on the command line.

The database server does not check the tape device when you specify the block size. Verify that the TAPEBLK tape device can read the block size that you specify. If not, you might not be able to read from the tape.

---

**TAPEDEV**

- **onconfig.std value**
  - On UNIX: /dev/tapedev
  - On Windows: \TAPE0
- **if not present**
  - On UNIX: /dev/null
- **units**
  - Pathname
- **takes effect**
  - For ontape: when you execute ontape
  - For onload and onunload: when the database server is shut down and restarted

**refer to**

- Using onload and onunload, in the IBM Informix Migration Guide
- Using ontape, in the IBM Informix Backup and Restore Guide
- "LTAPEDEV" on page 1-58

TAPEDEV specifies the device or directory file system to which ontape backs up storage spaces.

TAPEDEV also specifies the default device to which data is loaded or unloaded when you use the onload or onunload utilities. In Dynamic Server 10.0 and later, you can set TAPEDEV to STDIO to direct back up and restore operations to standard I/O instead of to a device.

If you change the tape device, verify that TAPEBLK and TAPESIZE are correct for the new device.

**Using Symbolic Links and a Remote Device (UNIX)**

TAPEDEV can be a symbolic link, enabling you to switch between tape devices without changing the pathname that TAPEDEV specifies.

Use the following syntax to specify a tape device attached to another host computer:

```
host_machine_name:tape_device_pathname
```

The following example specifies a tape device on the host computer kyoto:

```
kyoto:/dev/rmt01
```

**Rewinding Tape Devices Before Opening and on Closing**

The tape device that TAPEDEV specifies must perform a rewind before it opens and when it closes. The database server requires this action because of a series of checks that it performs before it writes to a tape.
When the database server attempts to write to any tape other than the first tape in a multivolume dbspace or logical-log backup, the database server first reads the tape header to make sure that the tape is available for use. Then the device is closed and reopened. The database server assumes the tape was rewound when it closed, and the database server begins to write.

Whenever the database server attempts to read a tape, it first reads the header and looks for the correct information. The database server does not find the correct header information at the start of the tape if the tape device did not rewind when it closed during the write process.

### TAPESIZE

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>10,240 units</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>range of values</strong></td>
<td>0 through 2,097,151</td>
</tr>
</tbody>
</table>
| **takes effect**       | For **ontape**: when you execute **ontape**  
                        | For **onload** and **onunload**: when the database server is shut down and restarted |
| **refer to**           | • Using **onload** and **onunload**, in the *IBM Informix Migration Guide*  
                        | • Using **ontape**, in the *IBM Informix Backup and Restore Guide*  
                        | • “LTAPESIZE” on page 1-59 |

**Note:** Tape size is irrelevant if TAPEDEV is set to STDIO.

The TAPESIZE parameter specifies the size of the device to which **ontape** backs up storage spaces. TAPESIZE also specifies the size of the default device to which data is loaded or unloaded when you use **onload** or **onunload**. If you are using **onload** or **onunload**, you can specify a different tape size on the command line. If you want to use the full physical capacity of a tape, set TAPESIZE to 0.

### TBLSPACE_STATS

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>1 if not present units</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>range of values</strong></td>
<td>0 or 1</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When the database server is shut down and restarted</td>
</tr>
</tbody>
</table>

The TBLSPACE_STATS configuration parameter turns on and off the collection of tblspace statistics. Use **onstat -g ppf** to list tblspace statistics.

To turn off the collection of tblspace statistics, set TBLSPACE_STATS to 0. When TBLSPACE_STATS is set to 0, **onstat -g ppf** displays “partition profiles disabled.” To turn on the collection of tblspace statistics, set TBLSPACE_STATS to 1.
**TBLTBLFIRST**

**onconfig.std value**

0

**units**

Kilobytes in multiples of page size

**range of values**

From the equivalent of 250 pages specified in kilobytes to the size of the first chunk minus the space needed for any system objects.

**takes effect**

When the database server is initialized

Specifies the first extent size of tblspace tblspace in the root dbspace. You might want to specify first and next extent sizes to reduce the number of tblspace tblspace extents and reduce the frequency of situations when you need to place the tblspace tblspace extents in non-primary chunks. (A primary chunk is the initial chunk in a dbspace.) For more information, see specifying first and next extent size in the chapter on managing dbspaces in the IBM Informix Administrator’s Guide.

You can use `oncheck -pt` and `oncheck -pT` to show the first and next extent sizes of a tblspace tblspace. For more information about the `oncheck` utility, see "oncheck -pt and -pT: Display tblspaces for a table or fragment” on page 7-18.

If you want to configure the first extent for a non-root dbspace, see information about the `onspaces` utility in Chapter 14, “The onspaces Utility,” on page 14-1.

**TBLTBLNEXT**

**onconfig.std value**

0

**units**

Kilobytes

**range of values**

From equivalent of 4 pages specified in kilobytes to the maximum chunk size minus three pages

**takes effect**

When the database server is initialized

Specifies the next extent size of tblspace tblspace in the root dbspace.

If there is not enough space for a next extent in the primary chunk, the extent is allocated from another chunk. If the specified space is not available, the closest available space is allocated. For more information on configuring extent sizes in tblspace tblspace, see “TBLTBLFIRST” on page 1-100.

**TEMPTAB_NOLOG**

**onconfig.std value**

0

**takes effect**

When the database server is shut down and restarted

**range of values**

0 = Enable logical logging on temporary table operations

1 = Disable logical logging on temporary table operations

**utilities**

`onmode -wf`
Use the TEMPTAB_NOLOG parameter to disable logging on temporary tables. This parameter can improve performance in application programs, especially in a data replication environment with HDR secondary, RS secondary, or SD secondary servers because it prevents Dynamic Server from transferring temporary tables over the network. The setting can be updated dynamically with the `onmode -wf` utility.

If you enable this setting, be aware that because no data is logged when using temporary tables, rolling back a transaction on a temporary table will no longer undo the work in the temporary table.

**TXTIMEOUT**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Units</th>
<th>Range of Values</th>
<th>Takes Effect</th>
<th>Refer To</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>300</td>
<td>Seconds</td>
<td>Positive integers</td>
<td>When the database server is shut down and restarted</td>
<td>How the two-phase commit protocol handles failures, in the chapter on multiphase commit protocols in the <em>IBM Informix Administrator’s Guide</em></td>
</tr>
</tbody>
</table>

TXTIMEOUT specifies the amount of time that a participant in a two-phase commit waits before it initiates participant recovery.

This parameter is used only for distributed queries that involve a remote database server. Nondistributed queries do not use this parameter.

**UNSECURE_ONSTAT**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Possible Values</th>
<th>Takes Effect</th>
<th>Refer To</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>not set</td>
<td>1 = All users can run <code>onstat</code> commands to view running SQL statements</td>
<td>When the database server is shut down and restarted</td>
<td><em>IBM Informix Administrator’s Guide</em></td>
</tr>
</tbody>
</table>

The `onstat` commands that show the SQL statement text that is executing on a session are by default normally restricted to DBSA users. To remove this restriction, set the UNSECURE_ONSTAT configuration parameter to 1. The `onstat` commands that show SQL statements include `onstat -g his`, `onstat -g ses`, `onstat -g stm`, `onstat -g ssc`, and `onstat -g sql`.

**USELASTCOMMITTED**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Range of Values</th>
<th>Takes Effect</th>
<th>Refer To</th>
</tr>
</thead>
<tbody>
<tr>
<td>onconfig.std</td>
<td>None</td>
<td>None = No isolation level identified</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Refer to "onmode -wf, -wm: Dynamically change certain configuration parameters" on page 11-20.
'Committed Read'
= All transactions from a Committed Read isolation level

'Dirty Read'
= All transactions from a Dirty Read isolation level

All = Both Committed Read and Dirty Read isolation levels

takes effect
When the database server is stopped and restarted.

utilities
onmode -wf or -wm

refer to
• 'onmode -wf, -wm: Dynamically change certain configuration parameters' on page 11-20
• SET ISOLATION statement information in IBM Informix Guide to SQL: Syntax
• Isolation level information in IBM Informix Performance Guide

USELASTCOMMITTED specifies the isolation level for which the LAST COMMITTED feature of the COMMITTED READ isolation level is implicitly in effect. The LAST COMMITTED feature can reduce the risk of locking conflicts between concurrent transactions on tables that have exclusive row locks. USELASTCOMMITTED can also enable LAST COMMITTED semantics for READ COMMITTED and READ UNCOMMITTED isolation levels of the SET TRANSACTION statement.

You can dynamically change the values of USELASTCOMMITTED through the SET ISOLATION statement or by using onmode -wf or onmode -wm.

USEOSTIME

onconfig.std value
0
range of values
0 = Off
1 = On
takes effect
During initialization
refer to
• Your IBM Informix Performance Guide
• Using the CURRENT function to return a datetime value, in the IBM Informix Guide to SQL: Syntax

Setting USEOSTIME to 1 specifies that the database server is to use subsecond precision when it obtains the current time from the operating system for SQL statements. The following example shows subseconds in a datetime value:
2001-09-29 12:50:04.612

If subsecond precision is not needed, the database server retrieves the current time from the operating system once per second, making the precision of time for client applications one second. If you set USEOSTIME to 0, the current function returns a zero (.000) for the year to fraction field.
When the host computer for the database server has a clock with subsecond precision, applications that depend on subsecond accuracy for their SQL statements should set USEOSTIME to 1.

Systems that run with USEOSTIME set to nonzero notice a performance degradation of up to 4 to 5 percent compared to running with USEOSTIME turned off.

This setting does not affect any calls regarding the time from application programs to Informix embedded-language library functions.

**VP_MEMORY_CACHE_KB**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>0 = Off</td>
</tr>
<tr>
<td></td>
<td>800 kilobytes per VP to the total kilobytes of the private caches in all VPs, not to exceed 40% of the memory limit as specified in the SHMTOTAL configuration parameter</td>
</tr>
<tr>
<td>takes effect</td>
<td>During initialization</td>
</tr>
<tr>
<td>utilities</td>
<td>onmode -wf or -wm</td>
</tr>
<tr>
<td></td>
<td>onstat -g vpcache</td>
</tr>
<tr>
<td>refer to</td>
<td>the CPU Virtual Processor Memory Caches section of the IBM Informix Performance Guide</td>
</tr>
</tbody>
</table>

The VP_MEMORY_CACHE_KB parameter enables the database server to access the private memory blocks of your CPU VP.

VP_MEMORY_CACHE_KB can be changed using the onmode -wf or -wm options; the changes take effect when the server is restarted. If the onmode options are set to 0, the memory caches are emptied. The onstat -g vpcache option returns information about CPU VP memory block cache statistics.

**VPCLASS**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>syntax</td>
<td>classname, options</td>
</tr>
<tr>
<td></td>
<td>The classname variable is required. Unlike most configuration parameters, VPCLASS has several option fields that can appear in any order, separated by commas. You cannot use any white space in the fields. VPCLASS has the following options:</td>
</tr>
<tr>
<td></td>
<td>num=num_VPs</td>
</tr>
<tr>
<td></td>
<td>max=max_VPs</td>
</tr>
<tr>
<td></td>
<td>aff=affinity</td>
</tr>
<tr>
<td></td>
<td>noage</td>
</tr>
<tr>
<td></td>
<td>noyield</td>
</tr>
</tbody>
</table>

For more information about using these options, refer to the individual discussions later in this section.
range of values

Up to 128 bytes. VPCLASS must be unique, begin with a letter or underscore, and contain only digits, letters, underscores, or $ characters.

takes effect

When the database server is shut down and restarted

utilities

onmode -p (to add or delete VP classes)

refer to

- Specifying user-defined classes of virtual processors, in the chapter on virtual processors in the IBM Informix Administrator’s Guide
- Specifying a nonyielding user-defined virtual processor (noyield option), in the chapter on virtual processors in the IBM Informix Administrator’s Guide
- Using onmode -p in “onmode -p: Add or remove virtual processors” on page 11-17
- “Using the noyield Option” on page 1-106
- IBM Informix User-Defined Routines and Data Types Developer’s Guide
- J/Foundation Developer’s Guide

The VPCLASS parameter allows you to designate a class of virtual processors (VPs), create a user-defined VP, and specify the following information for it:

- The number of virtual processors that the database server should start initially
- The maximum number of virtual processors allowed for this class
- The assignment of virtual processors to CPUs if processor affinity is available
- The disabling of priority aging by the operating system if the operating system implements priority aging

You can put several VPCLASS parameter definitions in your ONCONFIG file. Each VPCLASS parameter describes one class of virtual processors. Put each definition on a separate line, as in the following example:

```
VPCLASS cpu,num=8,aff=0-7,noage
VPCLASS new,num=0
```

### Default Values for the VPCLASS Options

The following table shows the defaults and value ranges for the VPCLASS parameter options.

<table>
<thead>
<tr>
<th>VPCLASS option</th>
<th>Class</th>
<th>Default Value</th>
<th>Range of Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>aio,num</td>
<td>AIO</td>
<td>1</td>
<td>1 to 10,000</td>
</tr>
<tr>
<td>cpu,num</td>
<td>CPU</td>
<td>1 if MULTIPROCESSOR is 0, 2 otherwise</td>
<td>1 to 10,000</td>
</tr>
<tr>
<td>num</td>
<td>All other classes</td>
<td>1</td>
<td>1 to 10,000</td>
</tr>
<tr>
<td>max_VPs</td>
<td>All</td>
<td>Unlimited</td>
<td>1 to 10,000</td>
</tr>
<tr>
<td>affinity</td>
<td>All</td>
<td>VPs are assigned to available processors in round-robin fashion.</td>
<td>Integers from 0 to (number of CPUs -1)</td>
</tr>
<tr>
<td>noage</td>
<td>All</td>
<td>Priority aging is in effect.</td>
<td>noage or omitted</td>
</tr>
</tbody>
</table>
Interaction of VPCLASS with Other Configuration Parameters

Using the VPCLASS parameter instead of the AFF_SPROC, AFF_NPROCS, NOAGE, NUMCPUVPS, and NUMAIOVPS parameters is required. If you use VPCLASS, you must explicitly remove other parameters from your ONCONFIG file. The following table shows the parameters that you must remove.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter to Remove</th>
</tr>
</thead>
<tbody>
<tr>
<td>VPCLASS cpu</td>
<td>NUMCPUVPS, AFF_SPROC, AFF_NPROCS, NOAGE</td>
</tr>
<tr>
<td>VPCLASS user-defined</td>
<td>SINGLE_CPU_VP</td>
</tr>
<tr>
<td>VPCLASS aio</td>
<td>NUMAIOVPS</td>
</tr>
</tbody>
</table>

VPCLASS Name

The first item in the VPCLASS parameter provides the name of the virtual-processor class that you are describing. The VPCLASS name is not case sensitive.

You can define new virtual-processor classes for user-defined routines or DataBlade modules, or you can set values for a predefined virtual-processor class. The following virtual-processor classes are predefined by the database server and have specific functions:

- adm
- lio
- shm
- adt
- msc
- soc
- cpu
- ntk
- str
- jvp
- opt
- tli
- kio
- aio
- pio
- encrypt

For VP classes tli, shm, str, and soc, you must set NETTYPE configuration parameter's VP_class field to NET.

For example, if the VPCLASS parameter is set as:

```
VPCLASS shm,num=1  
VPCLASS tli,num=1
```

then the NETTYPE parameter should be set as follows:

```
NETTYPE ipcshm,,3,NET  
NETTYPE ipctli,,3,NET
```

For more information on the NETTYPE configuration parameter, see "NETTYPE" on page 1-66.

The following example specifies that the database server should start three virtual processors of the CPU class:

```
VPCLASS cpu,num=3
```
Creating a User-Defined Class

The VPCLASS configuration parameter also allows you to create a class of user-defined virtual processors (VPs). A user-defined class of VPs can run ill-behaved user-defined routines (UDRs).

**Warning:** Execution of an ill-behaved routine in the CPU VP can cause serious interference with the operation of the database server. In addition, the routine itself might not produce correct results.

For more information on ill-behaved UDRs, see user-defined classes of virtual processors, in the chapter on virtual processors in the *IBM Informix Administrator's Guide*.

You might want to describe a user-defined class of virtual processors to run DataBlade or user-defined routines. The following example creates the user-defined class *new*, for which the database server starts three virtual processors initially:

```
VPCLASS new,num=3
```

At a later time, you can use `onmode -p` to add virtual processors to the class. The following command adds three virtual processors to the *new* class:

```
onmode -p +3 new
```

**Tip:** When you create a user-defined routine or function, you use the CLASS parameter of the CREATE FUNCTION statement to assign it to a class of virtual processors. You must ensure that the name of the user-defined class agrees with the name that you assigned in the CREATE FUNCTION statement. If you try to use a function that refers to a user-defined class, that class must exist and have virtual processors assigned to it. If the class does not have any virtual processors, you receive an SQL error.

For more information on how to assign a user-defined routine to either CPU or user-defined classes of virtual processors, refer to *IBM Informix User-Defined Routines and Data Types Developer's Guide*. For more information on the syntax of the CREATE FUNCTION or CREATE PROCEDURE statement, refer to the *IBM Informix Guide to SQL: Syntax*.

Using the noyield Option

By default, the VPCLASS parameter defines a yielding VP class, which allows the C UDR to yield to other threads that need access to the user-defined VP class. A UDR can perform blocking I/O calls if it executes in a yielding user-defined VP. However, it must still yield for other threads to have access to the VP.

You can also define nonyielding user-defined VPs with the `nocyield` option of VPCLASS. The `nocyield` option specifies creation of a nonyielding user-defined VP class. A nonyielding user-defined VP class executes a user-defined routine in a way
that gives the routine exclusive use of the virtual-processor class. In other words, user-defined routines that use a **noyield** virtual-processor class run serially. They never yield the VP to another thread.

You do not need to specify more than one VP in a nonyielding user-defined VP class, because the UDR runs on a single VP until it completes and any additional virtual processors would be idle.

**Important:** If your **UDR** uses global variables, only one VP in the user-defined virtual-processor class should be nonyielding.

The following example specifies a user-defined class of virtual processors called **new_noyield**, which runs in no-yield mode:

```plaintext
VPCLASS new_noyield,noyield,num=1
```

The **noyield** option applies only to user-defined VP classes. The database server ignores **noyield** if it is part of a VPCLASS parameter that defines a predefined VP class such as CPU, AIO, and so on.

### Using the num Option

The **num** option sets the number of virtual processors of the specified class that the database server should start during initialization.

On a single-processor computer, allocate only one CPU virtual processor. On a multiprocessor computer, do not allocate more CPU and user-defined virtual processors, combined, than there are CPUs on the computer.

Use the following syntax to specify the number of virtual processors:

```plaintext
num=num_VPs
```

### Specifying the Number of CPU VPs

For example, the following parameter specifies that the database server should start four virtual processors for the **cpu** class:

```plaintext
VPCLASS cpu,num=4
```

At a later time, you can use the **onmode -p** command to add virtual processors for the class.

### Using the max_VPs Option

The **max_VPs** option specifies the maximum number of virtual processors that the database server can start for the class.

Use the following syntax to specify the number of virtual processors:

```plaintext
max=max_VPs
```

The value can be any integer greater than 0. If you omit the **max_VPs** option, the number is unlimited.

### Using the affinity Option

On multiprocessor computers that support **processor affinity**, the affinity option specifies the CPUs to which the database server binds virtual processors.

The affinity option has the following two forms:
aff=processor_number
aff=start_range,end_range

In the first form, the database server binds all virtual processors in the class to the CPU numbered processor_number. (On a multiprocessor system, the operating system numbers the CPUs from 0 to (number of CPUs-1)). In the second form, the database server assigns the virtual processors of the class to processors in the range start_range to end_range, inclusive. The value end_range must be larger than start_range, and all values must be less than the total number of available CPUs.

For example, if your platform has eight CPUs, your ONCONFIG file might include the following VPCLASS entries:

```
VPCLASS    first,aff=3
VPCLASS    second,num=3,aff=5-7
VPCLASS    cpu,num=8,aff=0-7,noage
```

For more information about using processor affinity, refer to the chapter on virtual processors in the IBM Informix Administrator’s Guide.

**WSTATS**

- **onconfig.std**
  - **value**: 0
  - **range of values**: 0 = Disable wait statistics, 1 = Enable wait statistics
  - **takes effect**: When the database server is shut down and restarted

- **utilities**: onstat -g wst
- **refer to**: onstat -g wst: Print wait statistics for threads on page 15-114

WSTATS specifies the ability of onstat -g wst to print wait statistics for threads within the system. You should expect a small performance impact due to the cost of gathering statistical information and enabling WSTATS for production systems is not recommended.
Chapter 2. The sysmaster Database

In This Chapter

The sysmaster Database

The buildsmi Script

The bldutilsh Script

The System-Monitoring Interface

Understanding the SMI Tables

Accessing SMI Tables

SELECT Statements

Triggers and Event Alarms

SPL and SMI Tables

Locking and SMI Tables

The System-Monitoring Interface Tables

The sysutils Tables

sysadtnfo

sysaudit

syschkio

syscheckpoint

syschunks

sysconfig

sysdatabases

sysdbslocale

sysdbspaces

sysdri

sysdual

sysenv

sysemvses

sysextent

sysextentspaces

syslatency

sysl_type

sysl_workload

sysl

syslocks

syslogs

sysmgminfo

sysnetclienttype

sysnetglobal

sysnetworkio

sysonline

sysprofile

sysptprof

sysrslog

sysrscl

sysrsprof

sysrsessions

sysrm

sysrmses

sysrsqtrace

sysrsqtrace_info

sysrsqtrace_iter

sysrsrcc

sysrsrccs

sysrsrccds

sysstabnames

systthreads

systgrcss
In This Chapter

This chapter describes the **sysmaster** database and contains reference information for the *system-monitoring interface* (SMI). It provides information on the following topics:

- What is the **sysmaster** database
- How to use SMI tables
- Descriptions of the SMI tables
- A map of the documented SMI tables

For information about the ON–Bar tables, see the *IBM Informix Backup and Restore Guide*.

The sysmaster Database

The database server creates and maintains the **sysmaster** database. It is analogous to the system catalog for databases, which is described in the *IBM Informix Guide to SQL: Reference*. Just as a system catalog for every database managed by the database server keeps track of objects and privileges in the database, a **sysmaster** database for every database server keeps track of information about the database server.

The **sysmaster** database contains the *system-monitoring interface* (SMI) tables. The SMI tables provide information about the state of the database server. You can query these tables to identify processing bottlenecks, determine resource usage, track session or database server activity, and so on. This chapter describes these tables, which are slightly different from ordinary tables.

**Warning:** The database server relies on information in the **sysmaster** database. Do not change any of the tables in **sysmaster** or any of the data within the tables. Such changes could cause unpredictable and debilitating results.

The database server creates the **sysmaster** database when it initializes disk space. The database server creates the database with unbuffered logging. You cannot drop the database or any of the tables in it, and you cannot turn logging off.

As user **informix** on UNIX or a member of the **Informix-Admin** group on Windows, you can create SPL routines in the **sysmaster** database. (You can also create triggers on tables within **sysmaster**, but the database server never executes those triggers.)

Joins of multiple tables in **sysmaster** might return inconsistent results because the database server does not lock the tables during a join. You can join **sysmaster** tables with tables in other databases. However, to join **sysmaster** tables with tables in a nonlogging database, first make the nonlogging database the current database.

The **buildsmi** Script

When you bring the database server up for the first time, it runs a script called **buildsmi**, which is in the **etc** directory. This script builds the database and tables
that support SMI. The database server requires approximately 1750 free pages of logical-log space to build the sysmaster database.

If you receive an error message that directs you to run the buildsmi script, a problem probably occurred while the database server was building the SMI database, tables, and views. When you use buildsmi, the existing sysmaster database is dropped and then re-created.

**The bldutil.sh Script**

When you initialize the database server for the first time, it runs a script called bldutil.sh on UNIX or bldutil.bat on Windows. This script builds the sysutils database. If it fails, the database server creates an output file in the tmp directory. The output file is bldutil.process_id on UNIX and bldutil.out on Windows. The messages in this output file reflect errors that occurred during the script execution.

**The System-Monitoring Interface**

This section describes the SMI tables and how you access them to monitor the database server operation.

**Understanding the SMI Tables**

The SMI (system-monitoring interface) consists of tables and pseudo-tables that the database server maintains automatically. While the SMI tables appear to the user as tables, they are not recorded on disk as normal tables are. Instead, the database server constructs the tables in memory, on demand, based on information in shared memory at that instant. When you query an SMI table, the database server reads information from these shared-memory structures. Because the database server continually updates the data in shared memory, the information that SMI provides lets you examine the current state of your database server.

The SMI tables provide information about the following topics:

- Auditing
- Checkpoints
- Chunk I/O
- Chunks
- Database-logging status
- Dbspaces
- Disk usage
- Environment variables
- Extents
- Locks
- Networks
- SQL statement cache statistics
- SQL tracing
- System profiling
- Tables
- User profiling
- Virtual-processor CPU usage

The data in the SMI tables changes dynamically as users access and modify databases that the database server manages.
Accessing SMI Tables

Any user can use SQL SELECT statements to query an SMI table, but standard users cannot execute statements other than SELECT. Attempts to do so result in permission errors. The administrator can execute SQL statements other than SELECT, but the results of such statements are unpredictable.

Dynamic Server provides the sysadtinfo and sysaudit tables. Only user informix on UNIX or members of the Informix-Admin group on Windows can query sysadtinfo and sysaudit.

You cannot use dbschema or dbexport on any of the tables in the sysmaster database. If you do, the database server generates the following error message:

Database has pseudo tables - can't build schema

SELECT Statements

You can use SELECT statements on SMI tables wherever you can use SELECT against ordinary tables (from DB–Access, in an SPL routine, with ESQL/C, and so on) with one restriction: you cannot (meaningfully) reference rowid when you query SMI tables. SELECT statements that use rowid do not return an error, but the results are unpredictable.

All standard SQL syntax, including joins between tables, sorting of output, and so on, works with SMI tables. For example, if you want to join an SMI table with a non-SMI table, name the SMI table with the following standard syntax:

sysmaster[@dbname][:owner.]tablename

Triggers and Event Alarms

Triggers based on changes to SMI tables never run. Although you can define triggers on SMI tables, triggers are activated only when an INSERT, UPDATE, or DELETE statement occurs on a table. The updates to the SMI data occur within the database server, without the use of SQL, so a trigger on an SMI table is never activated, even though the data returned by a SELECT statement indicates that it should be.

To create an event alarm, query for a particular condition at predefined intervals, and execute an SPL routine if the necessary conditions for the alarm are met.

SPL and SMI Tables

You can access SMI tables from within a SPL routine. When you reference SMI tables, use the same syntax that you use to reference a standard table.

Locking and SMI Tables

The information in the SMI tables changes based on the database server activity. However, the database server does not update the information using SQL statements. When you use SMI tables with an isolation level that locks objects, it prevents other users from accessing the object, but it does not prevent the data from changing. In this sense, all the SMI tables have a permanent Dirty Read isolation level.

The System-Monitoring Interface Tables

The database server supports the following SMI tables.

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysadtinfo</td>
<td>Auditing configuration information</td>
<td>page 2-6</td>
</tr>
<tr>
<td>Table</td>
<td>Description</td>
<td>Reference</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>sysaudit</td>
<td>Auditing event masks</td>
<td>page 2-6</td>
</tr>
<tr>
<td>syscheckpoint</td>
<td>Checkpoint information</td>
<td>page 2-8</td>
</tr>
<tr>
<td>syschkio</td>
<td>Chunk I/O statistics</td>
<td>page 2-7</td>
</tr>
<tr>
<td>syschunks</td>
<td>Chunk information</td>
<td>page 2-8</td>
</tr>
<tr>
<td>sysconfig</td>
<td>Configuration information</td>
<td>page 2-9</td>
</tr>
<tr>
<td>sysdatabases</td>
<td>Database information</td>
<td>page 2-10</td>
</tr>
<tr>
<td>sybdblocale</td>
<td>Locale information</td>
<td>page 2-10</td>
</tr>
<tr>
<td>sydsbspaces</td>
<td>Dbspace information</td>
<td>page 2-12</td>
</tr>
<tr>
<td>syddi</td>
<td>Data-replication information</td>
<td>page 2-11</td>
</tr>
<tr>
<td>sysdual</td>
<td>Is a single-row table</td>
<td>page 2-12</td>
</tr>
<tr>
<td>sysenv</td>
<td>Online server’s startup environment</td>
<td>page 2-12</td>
</tr>
<tr>
<td>sysevnvs</td>
<td>Session-level environment variable</td>
<td>page 2-12</td>
</tr>
<tr>
<td>sysexents</td>
<td>Extent-allocation information</td>
<td>page 2-12</td>
</tr>
<tr>
<td>sysexxtspaces</td>
<td>External spaces information</td>
<td>page 2-12</td>
</tr>
<tr>
<td>sysha_latency</td>
<td>Secondary server latency statistics</td>
<td>page 2-13</td>
</tr>
<tr>
<td>sysha_type</td>
<td>Information about connected servers</td>
<td>page 2-13</td>
</tr>
<tr>
<td>sysha_workload</td>
<td>Secondary server workload statistics</td>
<td>page 2-14</td>
</tr>
<tr>
<td>sysipl</td>
<td>Index page logging information</td>
<td>page 2-15</td>
</tr>
<tr>
<td>syslocks</td>
<td>Active locks information</td>
<td>page 2-15</td>
</tr>
<tr>
<td>syslogs</td>
<td>Logical-log file information</td>
<td>page 2-16</td>
</tr>
<tr>
<td>syssmginfo</td>
<td>Memory Grant Manager/Parallel Data Query information</td>
<td>page 2-17</td>
</tr>
<tr>
<td>sysnetclienttype</td>
<td>Client type network activity</td>
<td>page 2-17</td>
</tr>
<tr>
<td>sysnetglobal</td>
<td>Global network information</td>
<td>page 2-17</td>
</tr>
<tr>
<td>sysnetworkio</td>
<td>Network I/O</td>
<td>page 2-18</td>
</tr>
<tr>
<td>sysonlineLog</td>
<td>Online log information</td>
<td>page 2-19</td>
</tr>
<tr>
<td>sysprofile</td>
<td>System-profile information</td>
<td>page 2-19</td>
</tr>
<tr>
<td>sysptprof</td>
<td>Table information</td>
<td>page 2-20</td>
</tr>
<tr>
<td>syssrsslog</td>
<td>RS secondary server information</td>
<td>page 2-21</td>
</tr>
<tr>
<td>sysscblst</td>
<td>Memory by user</td>
<td>page 2-21</td>
</tr>
<tr>
<td>syssesprof</td>
<td>Counts of various user actions</td>
<td>page 2-21</td>
</tr>
<tr>
<td>syssessions</td>
<td>Description of each user connected</td>
<td>page 2-22</td>
</tr>
<tr>
<td>syssmx</td>
<td>SMX (server multiplexer group) connection</td>
<td>page 2-23</td>
</tr>
<tr>
<td>syssmxses</td>
<td>SMX (server multiplexer group) session information</td>
<td>page 2-24</td>
</tr>
<tr>
<td>syssqltrace</td>
<td>SQL statement information</td>
<td>page 2-24</td>
</tr>
<tr>
<td>syssqltrace_info</td>
<td>SQL profile trace system information</td>
<td>page 2-25</td>
</tr>
<tr>
<td>syssqltrace_iter</td>
<td>SQL statement iterators</td>
<td>page 2-26</td>
</tr>
<tr>
<td>syssrcrss</td>
<td>RS secondary server statistics</td>
<td>page 2-26</td>
</tr>
<tr>
<td>syssrcsds</td>
<td>SD secondary server statistics</td>
<td>page 2-26</td>
</tr>
</tbody>
</table>
Many other tables in the **sysmaster** database are part of the system-monitoring interface but are not documented. Their schemas and column content can change from version to version. The **flags_text** table now contains more rows. To view the new rows you must first drop and then recreate the **sysmaster** database.

### The **sysutils** Tables
ON–Bar uses the following tables in the **sysutils** database. For more information, see the *IBM Informix Backup and Restore Guide*.

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>bar_action</strong></td>
<td>Lists all backup and restore actions that are attempted against an object, except during a cold restore. Use the information in this table to track backup and restore history.</td>
</tr>
<tr>
<td><strong>bar_instance</strong></td>
<td>Writes a record to this table for each successful backup. ON–Bar might later use the information for a restore operation.</td>
</tr>
<tr>
<td><strong>bar_object</strong></td>
<td>Describes each backup object. This table provides a list of all storage spaces and logical logs from each database server for which at least one backup attempt was made.</td>
</tr>
<tr>
<td><strong>bar_server</strong></td>
<td>Lists the database servers in an installation. This table is used to ensure that backup objects are returned to their proper places during a restore.</td>
</tr>
</tbody>
</table>

### **sysadtinfo**

The **sysadtinfo** table contains information about the auditing configuration for the database server. For more information, see your *IBM Informix Security Guide*. You must be user **informix** or user **root** on UNIX or a member of the **Informix-Admin** group on Windows to retrieve information from the **sysadtinfo** table.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>adtmode</td>
<td>integer</td>
<td>If auditing is on or off:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0 For off</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 For on</td>
</tr>
<tr>
<td>adterr</td>
<td>integer</td>
<td>Action on errors:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0 To continually retry audit writes until they succeed. Processing for the thread that generated the error stops.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 To write all failed audit writes to the message log and continue processing.</td>
</tr>
<tr>
<td>adtsize</td>
<td>integer</td>
<td>Maximum size of an audit file</td>
</tr>
<tr>
<td>adtpath</td>
<td>char(256)</td>
<td>Directory where audit files are written</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>adtfile</td>
<td>integer</td>
<td>Number of the audit file</td>
</tr>
</tbody>
</table>

**sysaudit**

For each defined audit mask (that is, for each *username*), the *sysaudit* table contains flags that represent the database events that generate audit records. The *success* and *failure* columns represent the bitmasks that compose the audit masks. If a bit is set in both the *success* and *failure* columns, the corresponding event generates an audit record whether or not the event succeeded.

You must be user *informix* or user *root* on UNIX or a member of the Informix-Admin group on Windows to retrieve information from the *sysaudit* table.

Use the *onaudit* utility to list or modify an audit mask. For information about *onaudit* and auditing, see your IBM Informix Security Guide.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>char(32)</td>
<td>Name of the mask</td>
</tr>
<tr>
<td>succ1</td>
<td>integer</td>
<td>Bitmask of the audit mask for success</td>
</tr>
<tr>
<td>succ2</td>
<td>integer</td>
<td>Bitmask of the audit mask for success</td>
</tr>
<tr>
<td>succ3</td>
<td>integer</td>
<td>Bitmask of the audit mask for success</td>
</tr>
<tr>
<td>succ4</td>
<td>integer</td>
<td>Bitmask of the audit mask for success</td>
</tr>
<tr>
<td>succ5</td>
<td>integer</td>
<td>Bitmask of the audit mask for success</td>
</tr>
<tr>
<td>fail1</td>
<td>integer</td>
<td>Bitmask of the audit mask for failure</td>
</tr>
<tr>
<td>fail2</td>
<td>integer</td>
<td>Bitmask of the audit mask for failure</td>
</tr>
<tr>
<td>fail3</td>
<td>integer</td>
<td>Bitmask of the audit mask for failure</td>
</tr>
<tr>
<td>fail4</td>
<td>integer</td>
<td>Bitmask of the audit mask for failure</td>
</tr>
<tr>
<td>fail5</td>
<td>integer</td>
<td>Bitmask of the audit mask for failure</td>
</tr>
</tbody>
</table>

**syschkio**

The *syschkio* table provides I/O statistics for individual chunks that the database server manages.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chunknum</td>
<td>smallint</td>
<td>Chunk number</td>
</tr>
<tr>
<td>reads</td>
<td>integer</td>
<td>Number of physical reads</td>
</tr>
<tr>
<td>pagesread</td>
<td>integer</td>
<td>Number of pages read</td>
</tr>
<tr>
<td>writes</td>
<td>integer</td>
<td>Number of physical writes</td>
</tr>
<tr>
<td>pageswritten</td>
<td>integer</td>
<td>Number of pages written</td>
</tr>
<tr>
<td>mreads</td>
<td>integer</td>
<td>Number of physical reads (mirror)</td>
</tr>
<tr>
<td>mpagesread</td>
<td>integer</td>
<td>Number of pages read (mirror)</td>
</tr>
<tr>
<td>mwrites</td>
<td>integer</td>
<td>Number of physical writes (mirror)</td>
</tr>
<tr>
<td>mpageswritten</td>
<td>integer</td>
<td>Number of pages written (mirror)</td>
</tr>
</tbody>
</table>
### syscheckpoint

The `syscheckpoint` table provides information and statistics about checkpoints.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>interval</td>
<td>integer</td>
<td>Number of checkpoints since the server was started</td>
</tr>
<tr>
<td>type</td>
<td>char(12)</td>
<td>Hard or Interval</td>
</tr>
<tr>
<td>caller</td>
<td>char(10)</td>
<td>Caller of the checkpoint</td>
</tr>
<tr>
<td>clock_time</td>
<td>integer</td>
<td>Time of day the checkpoint occurred</td>
</tr>
<tr>
<td>crit_time</td>
<td>float</td>
<td>Time spent waiting for the critical section to be released</td>
</tr>
<tr>
<td>flush_time</td>
<td>float</td>
<td>Time spent flushing pages to disk</td>
</tr>
<tr>
<td>cp_time</td>
<td>float</td>
<td>Duration from checkpoint pending until checkpoint done</td>
</tr>
<tr>
<td>n_dirty_buffs</td>
<td>integer</td>
<td>Number of dirty buffers</td>
</tr>
<tr>
<td>plogs_per_sec</td>
<td>integer</td>
<td>Number of physical log pages processed in a second</td>
</tr>
<tr>
<td>llogs_per_sec</td>
<td>integer</td>
<td>Number of logical log pages processed in a second</td>
</tr>
<tr>
<td>dskflush_per_sec</td>
<td>integer</td>
<td>Number of buffer pool pages flushed in a second</td>
</tr>
<tr>
<td>ckpt_logid</td>
<td>integer</td>
<td>Unique id of the logical log at the checkpoint</td>
</tr>
<tr>
<td>ckpt_logpos</td>
<td>integer</td>
<td>Position of the logical log at the checkpoint</td>
</tr>
<tr>
<td>physused</td>
<td>integer</td>
<td>Number of pages used in the physical log</td>
</tr>
<tr>
<td>logused</td>
<td>integer</td>
<td>Number of pages used in the logical log</td>
</tr>
<tr>
<td>n_crit_waits</td>
<td>integer</td>
<td>Number of users who had to wait to enter a critical section</td>
</tr>
<tr>
<td>tot_crit_wait</td>
<td>float</td>
<td>Duration spent waiting for all users waiting at the checkpoint critical section block</td>
</tr>
<tr>
<td>longest_crit_wait</td>
<td>float</td>
<td>Longest critical section wait</td>
</tr>
<tr>
<td>block_time</td>
<td>float</td>
<td>Duration of the checkpoint that blocked the system</td>
</tr>
</tbody>
</table>

### syschunks

The `syschunks` table describes each of the chunks that the database server manages. In the `flags` and `mflags` columns, each bit position represents a separate flag. Thus, it might be easier to read values in the `flags` and `mflags` columns if the values are returned using the HEX function.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chknum</td>
<td>smallint</td>
<td>Chunk number</td>
</tr>
<tr>
<td>dbsnum</td>
<td>smallint</td>
<td>Dbspace number</td>
</tr>
<tr>
<td>nxchknum</td>
<td>smallint</td>
<td>Number of the next chunk in this dbspace</td>
</tr>
<tr>
<td>chksize</td>
<td>integer</td>
<td>Number of pages in this chunk</td>
</tr>
<tr>
<td>offset</td>
<td>integer</td>
<td>Page offset of the chunk in its device or path</td>
</tr>
<tr>
<td>pagesize</td>
<td>integer</td>
<td>Page size (in units of system default page size)</td>
</tr>
<tr>
<td>nfree</td>
<td>integer</td>
<td>Number of free pages in the chunk (in units of system default page size)</td>
</tr>
<tr>
<td>is_offline</td>
<td>integer</td>
<td>1 If the chunk is offline, 0 if not</td>
</tr>
<tr>
<td>is_recovers</td>
<td>integer</td>
<td>1 If the chunk is being recovered, 0 if not</td>
</tr>
</tbody>
</table>
### Column Descriptions

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>is_blobchunk</td>
<td>integer</td>
<td>1 If the chunk is in a blobspace, 0 if not</td>
</tr>
<tr>
<td>is_sbcchk</td>
<td>integer</td>
<td>1 If the chunk is a sbchck, 0 if not</td>
</tr>
<tr>
<td>is_inconsistent</td>
<td>integer</td>
<td>1 If the chunk is undergoing logical restore, 0 if not</td>
</tr>
<tr>
<td>flags</td>
<td>smallint</td>
<td>Flags Hexadecimal Meaning</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16 0x0010 Chunk is a mirrored chunk</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32 0x0020 Chunk is in offline mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>64 0x0040 Chunk is in online mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128 0x0080 Chunk is in recovery mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>256 0x0100 Chunk has just been mirrored</td>
</tr>
<tr>
<td></td>
<td></td>
<td>512 0x0200 Chunk is part of a blobspace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1024 0x0400 Chunk is being dropped</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2048 0x0800 Chunk is part of an optical stageblob</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4096 0x1000 Chunk is inconsistent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16384 0x4000 Chunk contains temporary log space</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32768 0x8000 Chunk was added during roll forward</td>
</tr>
<tr>
<td>f_name</td>
<td>char(256)</td>
<td>Pathname for the file or device of this chunk</td>
</tr>
<tr>
<td>m_name</td>
<td>char(256)</td>
<td>Pathname for the file or device of the mirrored chunk, if any</td>
</tr>
<tr>
<td>m_offset</td>
<td>integer</td>
<td>Page offset of the mirrored chunk</td>
</tr>
<tr>
<td>mis_offline</td>
<td>integer</td>
<td>1 If mirror is offline, 0 if not</td>
</tr>
<tr>
<td>mis_recovering</td>
<td>integer</td>
<td>1 If mirror is being recovered, 0 if not</td>
</tr>
<tr>
<td>mflags</td>
<td>smallint</td>
<td>Mirrored chunk flags; values and meanings are the same as the flags column</td>
</tr>
</tbody>
</table>

### sysconfig

The `sysconfig` table describes the effective, original, and default values of the configuration parameters. For more information about the ONCONFIG file and the configuration parameters, see Chapter 1, “Configuration Parameters,” on page 1-1.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cf_id</td>
<td>integer</td>
<td>Unique numeric identifier</td>
</tr>
<tr>
<td>cf_name</td>
<td>char(128)</td>
<td>Configuration parameter name</td>
</tr>
<tr>
<td>cf_flags</td>
<td>integer</td>
<td>Reserved for future use</td>
</tr>
<tr>
<td>cf_original</td>
<td>char(256)</td>
<td>Value in the ONCONFIG file at boot time</td>
</tr>
<tr>
<td>cf_effective</td>
<td>char(256)</td>
<td>Value currently in use</td>
</tr>
<tr>
<td>cf_default</td>
<td>char(256)</td>
<td>Value provided by the database server if no value is specified in the ONCONFIG file</td>
</tr>
</tbody>
</table>
sysdatabases

The **sysdatabases** table describes each database that the database server manages.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>char(128)</td>
<td>Database name</td>
</tr>
<tr>
<td>partnum</td>
<td>integer</td>
<td>The partition number (tblspace identifier) for the systables table for the database</td>
</tr>
<tr>
<td>owner</td>
<td>char(32)</td>
<td>User ID of the creator of the database</td>
</tr>
<tr>
<td>created</td>
<td>date</td>
<td>Date created</td>
</tr>
<tr>
<td>is_logging</td>
<td>integer</td>
<td>1 If logging is active, 0 if not</td>
</tr>
<tr>
<td>is_buff_log</td>
<td>integer</td>
<td>1 If buffered logging, 0 if not</td>
</tr>
<tr>
<td>is_ansi</td>
<td>integer</td>
<td>1 If ANSI-compliant, 0 if not</td>
</tr>
<tr>
<td>is_nls</td>
<td>integer</td>
<td>1 If GLS-enabled, 0 if not</td>
</tr>
<tr>
<td>flags</td>
<td>smallint</td>
<td>Logging flags (hex values)</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>No logging</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Unbuffered logging</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Buffered logging</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>ANSI-compliant database</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>Read-only database</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>GLS database</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>Checking of the logging mode of syscdr database bypassed</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>Changed status to buffered logging</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>Changed status to unbuffered logging</td>
</tr>
<tr>
<td></td>
<td>400</td>
<td>Changed status to ANSI compliant</td>
</tr>
<tr>
<td></td>
<td>800</td>
<td>Database logging turned off</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>Long ID support enabled</td>
</tr>
</tbody>
</table>

sysdbslocale

The **sysdbslocale** table lists the locale of each database that the database server manages.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbs_dbsname</td>
<td>char(128)</td>
<td>Database name</td>
</tr>
<tr>
<td>dbs_collate</td>
<td>char(32)</td>
<td>The locale of the database</td>
</tr>
</tbody>
</table>

sysdbspaces

The **sysdbspaces** table describes each of the dbspaces that the database server manages. In the **flags** column, each bit position represents a separate flag. Thus, it might be easier to read values in the **flags** column if the values are returned using the HEX function.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbsnum</td>
<td>smallint</td>
<td>Dbspace number</td>
</tr>
<tr>
<td>name</td>
<td>char(128)</td>
<td>Dbspace name</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>owner</td>
<td>char(32)</td>
<td>User ID of owner of the dbspace</td>
</tr>
<tr>
<td>fchunk</td>
<td>smallint</td>
<td>Number of the first chunk in the dbspace</td>
</tr>
<tr>
<td>nchunks</td>
<td>smallint</td>
<td>Number of chunks in the dbspace</td>
</tr>
<tr>
<td>pagesize</td>
<td>integer</td>
<td>Page size</td>
</tr>
<tr>
<td>is_mirrored</td>
<td>integer</td>
<td>1 If dbspace is mirrored, 0 if not</td>
</tr>
<tr>
<td>is_blobspace</td>
<td>integer</td>
<td>1 If the dbspace is a blobspace, 0 if not</td>
</tr>
<tr>
<td>is_sbspace</td>
<td>integer</td>
<td>1 If the dbspace is a sbspace, 0 if not</td>
</tr>
<tr>
<td>is_temp</td>
<td>integer</td>
<td>1 If the dbspace is a temporary dbspace, 0 if not</td>
</tr>
<tr>
<td>flags</td>
<td>smallint</td>
<td>Flags</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Meaning</td>
</tr>
<tr>
<td>1</td>
<td>0x0001</td>
<td>Dbspace has no mirror</td>
</tr>
<tr>
<td>2</td>
<td>0x0002</td>
<td>Dbspace uses mirroring</td>
</tr>
<tr>
<td>4</td>
<td>0x0004</td>
<td>Dbspace mirroring is disabled</td>
</tr>
<tr>
<td>8</td>
<td>0x0008</td>
<td>Dbspace is newly mirrored</td>
</tr>
<tr>
<td>16</td>
<td>0x0010</td>
<td>Space is a blobspace</td>
</tr>
<tr>
<td>32</td>
<td>0x0020</td>
<td>Blobspace is on removable media</td>
</tr>
<tr>
<td>64</td>
<td>0x0040</td>
<td>Blobspace is on optical media</td>
</tr>
<tr>
<td>128</td>
<td>0x0080</td>
<td>Blobspace has been dropped.</td>
</tr>
<tr>
<td>256</td>
<td>0x0100</td>
<td>Blobspace is an optical stageblob</td>
</tr>
<tr>
<td>512</td>
<td>0x0200</td>
<td>Space is being recovered</td>
</tr>
<tr>
<td>1024</td>
<td>0x0400</td>
<td>Space has been physically recovered</td>
</tr>
<tr>
<td>2048</td>
<td>0x0800</td>
<td>Space is in logical recovery</td>
</tr>
<tr>
<td>32768</td>
<td>0x8000</td>
<td>Space is an sbspace</td>
</tr>
</tbody>
</table>

**sysdri**

The *sysdri* table provides information about the High-Availability Data-Replication status of the database server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>char(50)</td>
<td>High-Availability Data Replication type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• primary</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• secondary</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• standard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• not initialized</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>state</td>
<td>char(50)</td>
<td>State of High-Availability Data Replication Possible values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• off</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• on</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• connecting</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• failure</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• read-only</td>
</tr>
<tr>
<td>name</td>
<td>char(128)</td>
<td>The name of the other database server in the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>High-Availability Data-Replication pair</td>
</tr>
<tr>
<td>intvl</td>
<td>integer</td>
<td>The High-Availability Data-Replication interval</td>
</tr>
<tr>
<td>timeout</td>
<td>integer</td>
<td>The High-Availability Data-Replication timeout value for</td>
</tr>
<tr>
<td></td>
<td></td>
<td>this database server</td>
</tr>
<tr>
<td>lostfound</td>
<td>char(256)</td>
<td>The pathname to the lost-and-found file</td>
</tr>
</tbody>
</table>

**sysdual**

The *sysdual* table returns exactly one column and one row.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dummy</td>
<td>char(1)</td>
<td>Dummy columns returning &quot;X&quot;</td>
</tr>
</tbody>
</table>

**sysenv**

The *sysenv* table displays the startup environment settings of the database server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>env_id</td>
<td>integer</td>
<td>Identifier variable number</td>
</tr>
<tr>
<td>env_name</td>
<td>char(128)</td>
<td>Environment variable name</td>
</tr>
<tr>
<td>env_value</td>
<td>char(512)</td>
<td>Environment variable value</td>
</tr>
</tbody>
</table>

**sysenvses**

The *sysenvses* table displays the environment variable at the session level.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>envses_sid</td>
<td>integer</td>
<td>Session id</td>
</tr>
<tr>
<td>envses_id</td>
<td>integer</td>
<td>Identifier variable number</td>
</tr>
<tr>
<td>envses_name</td>
<td>char(128)</td>
<td>Session environment variable name</td>
</tr>
<tr>
<td>envses_value</td>
<td>char(512)</td>
<td>Session environment variable value</td>
</tr>
</tbody>
</table>

**sysextents**

The *sysextents* table provides information about extent allocation.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbname</td>
<td>char(128)</td>
<td>Database name</td>
</tr>
<tr>
<td>tabname</td>
<td>char(128)</td>
<td>Table name</td>
</tr>
<tr>
<td>chunk</td>
<td>integer</td>
<td>Chunk number</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>offset</td>
<td>integer</td>
<td>Number of pages into the chunk where the extent begins</td>
</tr>
<tr>
<td>size</td>
<td>integer</td>
<td>Size of the extent, in pages</td>
</tr>
</tbody>
</table>

**sysextspaces**

The sysextspaces table provides information about external spaces. Indexes for the id column and the name column allow only unique values.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>integer</td>
<td>External space ID</td>
</tr>
<tr>
<td>name</td>
<td>char(128)</td>
<td>External space name</td>
</tr>
<tr>
<td>owner</td>
<td>char(32)</td>
<td>External space owner</td>
</tr>
<tr>
<td>flags</td>
<td>integer</td>
<td>External space flags (reserved for future use)</td>
</tr>
<tr>
<td>refcnt</td>
<td>integer</td>
<td>External space reference count.</td>
</tr>
<tr>
<td>locsize</td>
<td>integer</td>
<td>Size of external space location, in bytes</td>
</tr>
<tr>
<td>location</td>
<td>char (256)</td>
<td>Location of external space</td>
</tr>
</tbody>
</table>

**syshta_latency**

The syshta_latency table provides a history of the amount of time that it took to apply a log record on any of the secondary nodes. This table contains a history of the last 20 samplings performed for a particular secondary server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lt_secondary</td>
<td>char(128)</td>
<td>Name of secondary server</td>
</tr>
<tr>
<td>lt_time_last_update</td>
<td>integer</td>
<td>Time at which log record was last updated</td>
</tr>
<tr>
<td>lt_lagtime_1</td>
<td>float</td>
<td>Amount of time required to apply log record for the most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_2</td>
<td>float</td>
<td>Amount of time required to apply log record for the second most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_3</td>
<td>float</td>
<td>Amount of time required to apply log record for the third most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_4</td>
<td>float</td>
<td>Amount of time required to apply log record for the fourth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_5</td>
<td>float</td>
<td>Amount of time required to apply log record for the fifth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_6</td>
<td>float</td>
<td>Amount of time required to apply log record for the sixth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_7</td>
<td>float</td>
<td>Amount of time required to apply log record for the seventh most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_8</td>
<td>float</td>
<td>Amount of time required to apply log record for the eighth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_9</td>
<td>float</td>
<td>Amount of time required to apply log record for the ninth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_10</td>
<td>float</td>
<td>Amount of time required to apply log record for the tenth most recent five-second interval</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>lt_lagtime_11</td>
<td>float</td>
<td>Amount of time required to apply log record for the eleventh most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_12</td>
<td>float</td>
<td>Amount of time required to apply log record for the twelfth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_13</td>
<td>float</td>
<td>Amount of time required to apply log record for the thirteenth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_14</td>
<td>float</td>
<td>Amount of time required to apply log record for the fourteenth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_15</td>
<td>float</td>
<td>Amount of time required to apply log record for the fifteenth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_16</td>
<td>float</td>
<td>Amount of time required to apply log record for the sixteenth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_17</td>
<td>float</td>
<td>Amount of time required to apply log record for the seventeenth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_18</td>
<td>float</td>
<td>Amount of time required to apply log record for the eighteenth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_19</td>
<td>float</td>
<td>Amount of time required to apply log record for the nineteenth most recent five-second interval</td>
</tr>
<tr>
<td>lt_lagtime_20</td>
<td>float</td>
<td>Amount of time required to apply log record for the twentieth most recent five-second interval</td>
</tr>
</tbody>
</table>

**sysha_type**

The `sysha_type` table is a single row table used to describe the type of server that is connected.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ha_type</td>
<td>integer</td>
<td>Server type (see table below)</td>
</tr>
<tr>
<td>ha_primary</td>
<td>char(128)</td>
<td>Server name (see table below)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value of ha_type</th>
<th>Value of ha_primary</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NULL</td>
<td>Not part of a high-availability environment</td>
</tr>
<tr>
<td>1</td>
<td>NULL</td>
<td>Primary server</td>
</tr>
<tr>
<td>2</td>
<td>&lt;primary server name&gt;</td>
<td>HDR secondary server</td>
</tr>
<tr>
<td>3</td>
<td>&lt;primary server name&gt;</td>
<td>SD secondary server</td>
</tr>
<tr>
<td>4</td>
<td>&lt;primary server name&gt;</td>
<td>RS secondary server</td>
</tr>
</tbody>
</table>

**sysha_workload**

The `sysha_workload` table contains workload statistics on each of the secondary servers.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wl_secondary</td>
<td>char(128)</td>
<td>Name of secondary server</td>
</tr>
<tr>
<td>wl_time_last_update</td>
<td>integer</td>
<td>Time at which workload last updated</td>
</tr>
</tbody>
</table>
### Column Description

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wl_type</td>
<td>char(12)</td>
<td>This row contains the ready queue size, user CPU time, and system CPU time</td>
</tr>
<tr>
<td>wl_workload_1</td>
<td>float</td>
<td>Most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_2</td>
<td>float</td>
<td>Second most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_3</td>
<td>float</td>
<td>Third most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_4</td>
<td>float</td>
<td>Fourth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_5</td>
<td>float</td>
<td>Fifth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_6</td>
<td>float</td>
<td>Sixth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_7</td>
<td>float</td>
<td>Seventh most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_8</td>
<td>float</td>
<td>Eighth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_9</td>
<td>float</td>
<td>Ninth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_10</td>
<td>float</td>
<td>Tenth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_11</td>
<td>float</td>
<td>Eleventh most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_12</td>
<td>float</td>
<td>Twelfth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_13</td>
<td>float</td>
<td>Thirteenth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_14</td>
<td>float</td>
<td>Fourteenth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_15</td>
<td>float</td>
<td>Fifteenth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_16</td>
<td>float</td>
<td>Sixteenth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_17</td>
<td>float</td>
<td>Seventeenth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_18</td>
<td>float</td>
<td>Eighteenth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_19</td>
<td>float</td>
<td>Nineteenth most recent workload activity</td>
</tr>
<tr>
<td>wl_workload_20</td>
<td>float</td>
<td>Twentieth most recent workload activity</td>
</tr>
</tbody>
</table>

### sysipl

The **sysipl** table provides information about the status of index page logging at the primary server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip1_status</td>
<td>integer</td>
<td>Index page logging status</td>
</tr>
<tr>
<td>ip1_time</td>
<td>integer</td>
<td>Time at which index page logging was enabled</td>
</tr>
</tbody>
</table>

### syslocks

The **syslocks** table provides information about all the currently active locks in the database server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbname</td>
<td>char(128)</td>
<td>Database name</td>
</tr>
<tr>
<td>tablename</td>
<td>char(128)</td>
<td>Table name</td>
</tr>
<tr>
<td>rowidlk</td>
<td>integer</td>
<td>Real rowid, if it is an index key lock</td>
</tr>
<tr>
<td>keynum</td>
<td>smallint</td>
<td>Key number of index key lock</td>
</tr>
</tbody>
</table>
### Column Type Description

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>char(4)</td>
<td>Type of lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B  Byte lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IS  Intent shared lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>S   Shared lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XS  Shared key value held by a repeatable reader</td>
</tr>
<tr>
<td></td>
<td></td>
<td>U   Update lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IX  Intent exclusive lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SIX  Shared intent exclusive lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>X   Exclusive lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XR  Exclusive key value held by a repeatable reader</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>integer</td>
<td>Session ID of the lock owner</td>
</tr>
<tr>
<td>waiter</td>
<td>integer</td>
<td>Session ID of the user waiting for the lock. If more than one user is waiting, only the first session ID appears.</td>
</tr>
</tbody>
</table>

### syslogs

The **syslogs** table provides information about space use in logical-log files. In the **flags** column, each bit position represents a separate flag. For example, for a log file, the **flags** column can have flags set for both current log file and temporary log file. Thus, it might be easier to read values in the **flags** column if the values are returned using the HEX function.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>smallint</td>
<td>Logical-log file number</td>
</tr>
<tr>
<td>uniqid</td>
<td>integer</td>
<td>Log-file ID</td>
</tr>
<tr>
<td>size</td>
<td>integer</td>
<td>Number of pages in the log file</td>
</tr>
<tr>
<td>used</td>
<td>integer</td>
<td>Number of pages used in the log file</td>
</tr>
<tr>
<td>is_used</td>
<td>integer</td>
<td>1 If file is used, 0 if not</td>
</tr>
<tr>
<td>is_current</td>
<td>integer</td>
<td>1 If file is the current file, 0 if not</td>
</tr>
<tr>
<td>is_backed_up</td>
<td>integer</td>
<td>1 If file has been backed up, 0 if not</td>
</tr>
<tr>
<td>is_new</td>
<td>integer</td>
<td>1 If the log has been added since the last level-0 dbspace backup, 0 if not</td>
</tr>
<tr>
<td>is_archived</td>
<td>integer</td>
<td>1 If file has been placed on the backup tape, 0 if not</td>
</tr>
<tr>
<td>is_temp</td>
<td>integer</td>
<td>1 If the file is flagged as a temporary log file, 0 if not</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flags</td>
<td>smallint</td>
<td>Flags</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32</td>
</tr>
</tbody>
</table>
The **sysmgm**info table provides an overview of the Memory Grant Manager (MGM) and Parallel Data Query (PDQ) information.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>max_query</td>
<td>integer</td>
<td>Maximum number of active queries allowed</td>
</tr>
<tr>
<td>total_mem</td>
<td>integer</td>
<td>Total MGM memory</td>
</tr>
<tr>
<td>avail_mem</td>
<td>integer</td>
<td>Free MGM memory</td>
</tr>
<tr>
<td>total_seq</td>
<td>integer</td>
<td>Total number of sequential scans</td>
</tr>
<tr>
<td>avail_seq</td>
<td>integer</td>
<td>Unused sequential scans</td>
</tr>
<tr>
<td>active</td>
<td>integer</td>
<td>Number of active MGM queries</td>
</tr>
<tr>
<td>ready</td>
<td>integer</td>
<td>Number of ready MGM queries</td>
</tr>
<tr>
<td>min_free_mem</td>
<td>integer</td>
<td>Minimum free MGM memory</td>
</tr>
<tr>
<td>avg_free_mem</td>
<td>float</td>
<td>Average free MGM memory</td>
</tr>
<tr>
<td>std_free_mem</td>
<td>float</td>
<td>Standard free MGM memory</td>
</tr>
<tr>
<td>min_free_seq</td>
<td>integer</td>
<td>Minimum free MGM sequential scans</td>
</tr>
<tr>
<td>avg_free_seq</td>
<td>float</td>
<td>Average free MGM sequential scans</td>
</tr>
<tr>
<td>std_free_seq</td>
<td>float</td>
<td>Standard free MGM sequential scans</td>
</tr>
<tr>
<td>max_active</td>
<td>integer</td>
<td>Maximum active MGM SQL operations</td>
</tr>
<tr>
<td>cnt_active</td>
<td>integer</td>
<td>Number of active MGM SQL operations</td>
</tr>
<tr>
<td>avg_active</td>
<td>float</td>
<td>Maximum active MGM SQL operations</td>
</tr>
<tr>
<td>std_active</td>
<td>float</td>
<td>Standard active MGM SQL operations</td>
</tr>
<tr>
<td>max_ready</td>
<td>integer</td>
<td>Maximum ready MGM SQL operations</td>
</tr>
<tr>
<td>cnt_ready</td>
<td>integer</td>
<td>Number of ready MGM SQL operations</td>
</tr>
<tr>
<td>avg_ready</td>
<td>float</td>
<td>Average ready MGM SQL operations</td>
</tr>
<tr>
<td>std_ready</td>
<td>float</td>
<td>Standard ready MGM SQL operations</td>
</tr>
</tbody>
</table>

The **sysnetclienttype** table provides an overview of the network activity for each client type.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nc_cons_allowed</td>
<td>integer</td>
<td>Whether or not connections are allowed</td>
</tr>
<tr>
<td>nc_accepted</td>
<td>integer</td>
<td>Number of connections that were accepted</td>
</tr>
<tr>
<td>nc_rejected</td>
<td>integer</td>
<td>Number of network connections that were rejected</td>
</tr>
<tr>
<td>nc_reads</td>
<td>int8</td>
<td>Number of network reads for this client type</td>
</tr>
<tr>
<td>nc_writes</td>
<td>int8</td>
<td>Number of network writes for this client type</td>
</tr>
<tr>
<td>nc_name</td>
<td>char(18)</td>
<td>Name of the client type</td>
</tr>
</tbody>
</table>

The **sysnetglobal** table provides an overview of the system network.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ng_reads</td>
<td>int8</td>
<td>Number of network reads</td>
</tr>
</tbody>
</table>
### Column Type Description

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ng_writes</strong></td>
<td>int8</td>
<td>Number of network writes</td>
</tr>
<tr>
<td><strong>ng_connects</strong></td>
<td>int8</td>
<td>Number of network connections</td>
</tr>
<tr>
<td><strong>ng_his_read_count</strong></td>
<td>int8</td>
<td>Number of network reads by users who have disconnected ng_his_read_bytes</td>
</tr>
<tr>
<td><strong>ng_his_read_bytes</strong></td>
<td>int8</td>
<td>Data transferred to the server by users who have disconnected</td>
</tr>
<tr>
<td><strong>ng_his_write_count</strong></td>
<td>int8</td>
<td>Number of network writes by users who have disconnected</td>
</tr>
<tr>
<td><strong>ng_his_write_bytes</strong></td>
<td>int8</td>
<td>Data transferred to the client by users who have disconnected</td>
</tr>
<tr>
<td><strong>ng_num_netscb</strong></td>
<td>integer</td>
<td>Number of network subscribers</td>
</tr>
<tr>
<td><strong>ng_max_netscb</strong></td>
<td>integer</td>
<td>Maximum number of network subscribers</td>
</tr>
<tr>
<td><strong>ng_free_thres</strong></td>
<td>integer</td>
<td>Threshold for the maximum number of freed buffers in the buffer list</td>
</tr>
<tr>
<td><strong>ng_free_cnt</strong></td>
<td>integer</td>
<td>Number of times the ng_free_thres limit has been reached</td>
</tr>
<tr>
<td><strong>ng_wait_thres</strong></td>
<td>integer</td>
<td>Threshold for the maximum number of buffers that can be held in the buffer list for one connection</td>
</tr>
<tr>
<td><strong>ng_wait_cnt</strong></td>
<td>integer</td>
<td>Number of times the ng_wait_thres limit has been reached</td>
</tr>
<tr>
<td><strong>ng_pvt_thres</strong></td>
<td>integer</td>
<td>Threshold for the maximum number of freed buffers in the private buffer queue</td>
</tr>
<tr>
<td><strong>ng_netbuf_size</strong></td>
<td>integer</td>
<td>Size of the transport network buffers</td>
</tr>
<tr>
<td><strong>ng_buf_alloc</strong></td>
<td>integer</td>
<td>Number of network buffers allocated</td>
</tr>
<tr>
<td><strong>ng_buf_alloc_max</strong></td>
<td>integer</td>
<td>Maximum value of allocated network buffers</td>
</tr>
<tr>
<td><strong>ng_netscb_id</strong></td>
<td>integer</td>
<td>Next netscb id</td>
</tr>
</tbody>
</table>

### sysnetworkio

The **sysnetglobal** table provides an overview of the system network.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>net_id</strong></td>
<td>integer</td>
<td>Netscb id</td>
</tr>
<tr>
<td><strong>net_sid</strong></td>
<td>integer</td>
<td>Session id</td>
</tr>
<tr>
<td><strong>net_netscb</strong></td>
<td>int8</td>
<td>Netscb prt</td>
</tr>
<tr>
<td><strong>net_client_type</strong></td>
<td>integer</td>
<td>Client type Int</td>
</tr>
<tr>
<td><strong>net_client_name</strong></td>
<td>char(12)</td>
<td>Client protocol name</td>
</tr>
<tr>
<td><strong>net_read_cnt</strong></td>
<td>int8</td>
<td>Number of network reads</td>
</tr>
<tr>
<td><strong>net_write_cnt</strong></td>
<td>int8</td>
<td>Number of network writes</td>
</tr>
<tr>
<td><strong>net_open_time</strong></td>
<td>integer</td>
<td>Time this session connected</td>
</tr>
<tr>
<td><strong>net_last_read</strong></td>
<td>integer</td>
<td>Time of the last read from the network</td>
</tr>
<tr>
<td><strong>net_last_write</strong></td>
<td>integer</td>
<td>Time of the last write from the network</td>
</tr>
<tr>
<td><strong>net_stage</strong></td>
<td>integer</td>
<td>Connect / Disconnect / Receive</td>
</tr>
<tr>
<td><strong>net_options</strong></td>
<td>integer</td>
<td>Options from sqhosts</td>
</tr>
<tr>
<td><strong>net_protocol</strong></td>
<td>integer</td>
<td>Protocol</td>
</tr>
</tbody>
</table>
### Column Types

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>net_type</td>
<td>char(10)</td>
<td>Type of network protocol</td>
</tr>
<tr>
<td>net_server_fd</td>
<td>integer</td>
<td>Server fd</td>
</tr>
<tr>
<td>net_poll_thread</td>
<td>integer</td>
<td>Poll thread</td>
</tr>
</tbody>
</table>

### sysonlinelog

The `sysonlinelog` table provides a view of the information stored in the `online.log` file.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>offset</td>
<td>int8</td>
<td>File offset</td>
</tr>
<tr>
<td>next_offset</td>
<td>int8</td>
<td>Offset to the next message</td>
</tr>
<tr>
<td>line</td>
<td>char(4096)</td>
<td>Single line of text from the file</td>
</tr>
</tbody>
</table>

### sysprofile

The `sysprofile` table contains profile information about the database server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>char(13)</td>
<td>Name of profiled event. (See table that follows for a list of possible events.)</td>
</tr>
<tr>
<td>value</td>
<td>integer</td>
<td>Value of profiled event. (See table that follows for a list of possible events.)</td>
</tr>
</tbody>
</table>

### Events Profiled in sysprofile

The following table lists the events that, together with a corresponding value, make up the rows of the `sysprofile` table.

<table>
<thead>
<tr>
<th>Events Profiled in sysprofile</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dskreads</td>
<td>Number of actual reads from disk</td>
</tr>
<tr>
<td>bufreads</td>
<td>Number of reads from shared memory</td>
</tr>
<tr>
<td>dskwrites</td>
<td>Actual number of writes to disk</td>
</tr>
<tr>
<td>bufwrites</td>
<td>Number of writes to shared memory</td>
</tr>
<tr>
<td>isamtot</td>
<td>Total number of calls</td>
</tr>
<tr>
<td>isopens</td>
<td>isopen calls</td>
</tr>
<tr>
<td>issstarts</td>
<td>issstart calls</td>
</tr>
<tr>
<td>isreads</td>
<td>isread calls</td>
</tr>
<tr>
<td>iswwrites</td>
<td>iswrite calls</td>
</tr>
<tr>
<td>isrewrites</td>
<td>isrewrite calls</td>
</tr>
<tr>
<td>isdeletes</td>
<td>isdelete calls</td>
</tr>
<tr>
<td>iscommits</td>
<td>iscommit calls</td>
</tr>
<tr>
<td>isrollbacks</td>
<td>isrollback calls</td>
</tr>
<tr>
<td>ovlock</td>
<td>Overflow lock table</td>
</tr>
<tr>
<td>ovuser</td>
<td>Overflow user table</td>
</tr>
<tr>
<td>ovtrans</td>
<td>Overflow transaction table</td>
</tr>
<tr>
<td>Events Profiled in sysprofile</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>latchwts</td>
<td>Latch request waits</td>
</tr>
<tr>
<td>bufwts</td>
<td>Buffer waits</td>
</tr>
<tr>
<td>lockreqs</td>
<td>Lock requests</td>
</tr>
<tr>
<td>lockwts</td>
<td>Lock waits</td>
</tr>
<tr>
<td>ckptwts</td>
<td>Checkpoint waits</td>
</tr>
<tr>
<td>deadlks</td>
<td>Deadlocks</td>
</tr>
<tr>
<td>lktouts</td>
<td>Deadlock time-outs</td>
</tr>
<tr>
<td>numckpts</td>
<td>Number checkpoints</td>
</tr>
<tr>
<td>plgpagewrites</td>
<td>Physical-log pages written</td>
</tr>
<tr>
<td>plgwrites</td>
<td>Physical-log writes</td>
</tr>
<tr>
<td>llgrecs</td>
<td>Logical-log records</td>
</tr>
<tr>
<td>llgpagewrites</td>
<td>Logical-log writes</td>
</tr>
<tr>
<td>llgwrites</td>
<td>Logical-log pages written</td>
</tr>
<tr>
<td>pagreads</td>
<td>Page reads</td>
</tr>
<tr>
<td>pagwrites</td>
<td>Page writes</td>
</tr>
<tr>
<td>flushes</td>
<td>Buffer-pool flushes</td>
</tr>
<tr>
<td>compress</td>
<td>Page compresses</td>
</tr>
<tr>
<td>fgwrites</td>
<td>Foreground writes</td>
</tr>
<tr>
<td>lruwrites</td>
<td>Least-recently used (LRU) writes</td>
</tr>
<tr>
<td>chunkwrites</td>
<td>Writes during a checkpoint</td>
</tr>
<tr>
<td>btradata</td>
<td>Read-ahead data pages read through index leaf node</td>
</tr>
<tr>
<td>btraidx</td>
<td>Read-ahead data pages read through index branch or root node</td>
</tr>
<tr>
<td>dpra</td>
<td>Data pages read into memory with read-ahead feature</td>
</tr>
<tr>
<td>rapg_us_used</td>
<td>Read-ahead data pages that user used</td>
</tr>
<tr>
<td>seqscans</td>
<td>Sequential scans</td>
</tr>
<tr>
<td>totalsorts</td>
<td>Total sorts</td>
</tr>
<tr>
<td>memsorts</td>
<td>Sorts that fit in memory</td>
</tr>
<tr>
<td>disksorts</td>
<td>Sorts that did not fit in memory</td>
</tr>
<tr>
<td>maxsortspace</td>
<td>Maximum disk space used by a sort</td>
</tr>
</tbody>
</table>

**sysptprof**

The `sysptprof` table lists information about a tblspace. Tblspaces correspond to tables.

Profile information for a table is available only when a table is open. When the last user who has a table open closes it, the tblspace in shared memory is freed, and any profile statistics are lost.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbname</td>
<td>char(128)</td>
<td>Database name</td>
</tr>
<tr>
<td>tabname</td>
<td>char(128)</td>
<td>Table name</td>
</tr>
<tr>
<td>partnum</td>
<td>integer</td>
<td>Partition (tblspace) number</td>
</tr>
</tbody>
</table>
The `sysrsslog` table captures information about RS secondary servers at the primary server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>server_name</td>
<td>char(128)</td>
<td>Server name</td>
</tr>
<tr>
<td>from_cache</td>
<td>integer</td>
<td>Total pages read from cache</td>
</tr>
<tr>
<td>from_disk</td>
<td>integer</td>
<td>Total pages read from disk</td>
</tr>
<tr>
<td>logpages_tossed</td>
<td>integer</td>
<td>Total number of log pages not written to log buffer cache</td>
</tr>
</tbody>
</table>

The columns of the `sysscblst` table provide information about session memory amounts.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>memtotal</td>
<td>integer</td>
<td>Total memory available</td>
</tr>
<tr>
<td>memused</td>
<td>integer</td>
<td>Total memory used</td>
</tr>
</tbody>
</table>

The `syssesprof` table lists cumulative counts of the number of occurrences of user actions such as writes, deletes, or commits.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
<td>integer</td>
<td>Session ID</td>
</tr>
<tr>
<td>lockreqs</td>
<td>integer</td>
<td>Number of locks requested</td>
</tr>
<tr>
<td>lockheld</td>
<td>integer</td>
<td>Number of locks currently held</td>
</tr>
<tr>
<td>lockwts</td>
<td>integer</td>
<td>Number of times waited for a lock</td>
</tr>
<tr>
<td>deadlks</td>
<td>integer</td>
<td>Number of deadlocks detected</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-----------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>lktouts</td>
<td>smallint</td>
<td>Number of deadlock timeouts</td>
</tr>
<tr>
<td>logrecs</td>
<td>integer</td>
<td>Number of logical-log records written</td>
</tr>
<tr>
<td>isreads</td>
<td>integer</td>
<td>Number of reads</td>
</tr>
<tr>
<td>iswrites</td>
<td>integer</td>
<td>Number of writes</td>
</tr>
<tr>
<td>isrewrites</td>
<td>integer</td>
<td>Number of rewrites</td>
</tr>
<tr>
<td>isdeletes</td>
<td>integer</td>
<td>Number of deletes</td>
</tr>
<tr>
<td>iscommits</td>
<td>integer</td>
<td>Number of commits</td>
</tr>
<tr>
<td>isrollbacks</td>
<td>integer</td>
<td>Number of rollbacks</td>
</tr>
<tr>
<td>longtxs</td>
<td>integer</td>
<td>Number of long transactions</td>
</tr>
<tr>
<td>bufreads</td>
<td>integer</td>
<td>Number of buffer reads</td>
</tr>
<tr>
<td>bufwrites</td>
<td>integer</td>
<td>Number of buffer writes</td>
</tr>
<tr>
<td>seqscans</td>
<td>integer</td>
<td>Number of sequential scans</td>
</tr>
<tr>
<td>pagreads</td>
<td>integer</td>
<td>Number of page reads</td>
</tr>
<tr>
<td>pagwrites</td>
<td>integer</td>
<td>Number of page writes</td>
</tr>
<tr>
<td>total_sorts</td>
<td>integer</td>
<td>Number of total sorts</td>
</tr>
<tr>
<td>dksorts</td>
<td>integer</td>
<td>Number of sorts that did not fit in memory</td>
</tr>
<tr>
<td>max_sortdisksp</td>
<td>integer</td>
<td>Maximum space used by a sort</td>
</tr>
<tr>
<td>logspused</td>
<td>integer</td>
<td>Number of bytes of logical-log space used by current transaction of session</td>
</tr>
<tr>
<td>maxlogsp</td>
<td>integer</td>
<td>Maximum number of bytes of logical-log space ever used by the session</td>
</tr>
</tbody>
</table>

**syssessions**

The **syssessions** table provides general information on each user connected to the database server. In the **state** column, each bit position represents a separate flag. Thus, it might be easier to read values in the **state** column if the values are returned using the HEX function.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sid</td>
<td>integer</td>
<td>Session ID</td>
</tr>
<tr>
<td>username</td>
<td>char(32)</td>
<td>User ID</td>
</tr>
<tr>
<td>uid</td>
<td>smallint</td>
<td>User ID number</td>
</tr>
<tr>
<td>pid</td>
<td>integer</td>
<td>Process ID of the client</td>
</tr>
<tr>
<td>hostname</td>
<td>char(16)</td>
<td>Hostname of client</td>
</tr>
<tr>
<td>tty</td>
<td>char(16)</td>
<td>Name of the user’s stderr file</td>
</tr>
<tr>
<td>connected</td>
<td>integer</td>
<td>Time that user connected to the database server</td>
</tr>
<tr>
<td>feprogram</td>
<td>char(16)</td>
<td>Reserved for future use</td>
</tr>
<tr>
<td>pooladdr</td>
<td>integer</td>
<td>Session pool address</td>
</tr>
<tr>
<td>is_wlatch</td>
<td>integer</td>
<td>1 If the primary thread for the session is waiting for a latch</td>
</tr>
<tr>
<td>is_wlock</td>
<td>integer</td>
<td>1 If the primary thread for the session is waiting for a lock</td>
</tr>
<tr>
<td>is_wbuff</td>
<td>integer</td>
<td>1 If the primary thread for the session is waiting for a buffer</td>
</tr>
<tr>
<td>is_wckpt</td>
<td>integer</td>
<td>1 If the primary thread for the session is waiting for a checkpoint</td>
</tr>
</tbody>
</table>
### Column Type Description

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>is_wlogbuf</td>
<td>integer</td>
<td>1 If the primary thread for the session is waiting for a log buffer</td>
</tr>
<tr>
<td>is_wtrans</td>
<td>integer</td>
<td>1 If the primary thread for the session is waiting for a transaction</td>
</tr>
<tr>
<td>is_monitor</td>
<td>integer</td>
<td>1 If the session is a special monitoring process</td>
</tr>
<tr>
<td>is_incrit</td>
<td>integer</td>
<td>1 If the primary thread for the session is in a critical section</td>
</tr>
<tr>
<td>state</td>
<td>integer</td>
<td>Flags Hexadecimal Meaning</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 0x00000001 User structure in use</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 0x00000002 Waiting for a latch</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 0x00000004 Waiting for a lock</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 0x00000008 Waiting for a buffer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16 0x00000010 Waiting for a checkpoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32 0x00000020 In a read call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>64 0x00000040 Writing logical-log file to backup tape</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128 0x00000080 ON-Monitor (UNIX)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>256 0x00000100 In a critical section</td>
</tr>
<tr>
<td></td>
<td></td>
<td>512 0x00000200 Special daemon</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1024 0x00000400 Archiving</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2048 0x00000800 Clean up dead processes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4096 0x00001000 Waiting for write of log buffer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8192 0x00002000 Special buffer-flushing thread</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16384 0x00004000 Remote database server</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32768 0x00008000 Deadlock timeout used to set RS_timeout</td>
</tr>
<tr>
<td></td>
<td></td>
<td>65536 0x00010000 Regular lock timeout</td>
</tr>
<tr>
<td></td>
<td></td>
<td>262144 0x00040000 Waiting for a transaction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>524288 0x00080000 Primary thread for a session</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1048576 0x00100000 Thread for building indexes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2097152 0x00200000 B-tree cleaner thread</td>
</tr>
</tbody>
</table>

### syssmx

The syssmx table provides SMX (server multiplexer group) connection information.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>int8</td>
<td>SMX pipe address</td>
</tr>
<tr>
<td>name</td>
<td>char(128)</td>
<td>Target server name</td>
</tr>
<tr>
<td>encryption_status</td>
<td>char(20)</td>
<td>Enabled or disabled</td>
</tr>
<tr>
<td>buffers_sent</td>
<td>Integer</td>
<td>Number of buffers sent</td>
</tr>
<tr>
<td>buffers_recv</td>
<td>Integer</td>
<td>Number of buffers received</td>
</tr>
<tr>
<td>bytes_sent</td>
<td>int8</td>
<td>Number of bytes sent</td>
</tr>
<tr>
<td>bytes_recv</td>
<td>int8</td>
<td>Number of bytes received</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>--------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>reads</td>
<td>integer</td>
<td>Number of read calls</td>
</tr>
<tr>
<td>writes</td>
<td>integer</td>
<td>Number of write calls</td>
</tr>
<tr>
<td>retries</td>
<td>integer</td>
<td>Number of write call retries</td>
</tr>
</tbody>
</table>

**syssmxses**

The `syssmxses` table provides SMX (server multiplexer group) session information.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>char(128)</td>
<td>Target server name</td>
</tr>
<tr>
<td>address</td>
<td>int8</td>
<td>SMX session address</td>
</tr>
<tr>
<td>client_type</td>
<td>char(20)</td>
<td>SMX client type</td>
</tr>
<tr>
<td>reads</td>
<td>integer</td>
<td>Number of read calls</td>
</tr>
<tr>
<td>writes</td>
<td>integer</td>
<td>Number of write calls</td>
</tr>
</tbody>
</table>

**syssqltrace**

The `syssqltrace` table provides detailed information about a single SQL statement.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sql_id</td>
<td>int8</td>
<td>Unique SQL execution ID</td>
</tr>
<tr>
<td>sql_address</td>
<td>int8</td>
<td>Address of the statement in the code block</td>
</tr>
<tr>
<td>sql_sid</td>
<td>int</td>
<td>Database session ID of the user running the SQL statement</td>
</tr>
<tr>
<td>sql_uid</td>
<td>int</td>
<td>User ID of the statement running the SQL</td>
</tr>
<tr>
<td>sql_stmttype</td>
<td>int</td>
<td>Statement type</td>
</tr>
<tr>
<td>sql_stmtname</td>
<td>char(40)</td>
<td>Statement type displayed as a word</td>
</tr>
<tr>
<td>sql_finishtime</td>
<td>int</td>
<td>Time this statement completed (UNIX)</td>
</tr>
<tr>
<td>sql_begintxtime</td>
<td>int</td>
<td>Time this transaction started</td>
</tr>
<tr>
<td>sql_runtime</td>
<td>float</td>
<td>Statement execution time</td>
</tr>
<tr>
<td>sql_pgreads</td>
<td>int</td>
<td>Number of disk reads for this SQL statement</td>
</tr>
<tr>
<td>sql_bfreads</td>
<td>int</td>
<td>Number of buffer reads for this SQL statement</td>
</tr>
<tr>
<td>sql_rdcache</td>
<td>float</td>
<td>Percentage of time the page was read from the buffer pool</td>
</tr>
<tr>
<td>sql_bfidxreads</td>
<td>int</td>
<td>Number of index page buffer reads</td>
</tr>
<tr>
<td>sql_pgwrites</td>
<td>int</td>
<td>Number of pages written to disk</td>
</tr>
<tr>
<td>sql_bfwrites</td>
<td>int</td>
<td>Number of pages modified and returned to the buffer pool</td>
</tr>
<tr>
<td>sql_wrcache</td>
<td>float</td>
<td>Percentage of time a page was written to the buffer pool but not to disk</td>
</tr>
<tr>
<td>sql_lockreq</td>
<td>int</td>
<td>Total number of locks required by this SQL statement</td>
</tr>
<tr>
<td>sql_lockwaits</td>
<td>int</td>
<td>Number of times the SQL statement waited on locks</td>
</tr>
<tr>
<td>sql_lockwtttime</td>
<td>float</td>
<td>Time the system waited for locks during SQL statement</td>
</tr>
<tr>
<td>sql_logsparse</td>
<td>int</td>
<td>Amount of space the SQL statement used in the logical log</td>
</tr>
</tbody>
</table>
### Column Types

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sql_sorttotal</td>
<td>int</td>
<td>Number of sorts that ran for the statement</td>
</tr>
<tr>
<td>sql_sortdisk</td>
<td>int</td>
<td>Number of sorts that ran on disk</td>
</tr>
<tr>
<td>sql_sortmem</td>
<td>int</td>
<td>Number of sorts that ran in memory</td>
</tr>
<tr>
<td>sql_executions</td>
<td>int</td>
<td>Number of times the SQL statement ran</td>
</tr>
<tr>
<td>sql_totaltime</td>
<td>float</td>
<td>Total amount of time spent running the statement</td>
</tr>
<tr>
<td>sql_avgtime</td>
<td>float</td>
<td>Average amount of time spent running the statement</td>
</tr>
<tr>
<td>sql_maxtime</td>
<td>float</td>
<td>Maximum amount of time spent executing the SQL statement</td>
</tr>
<tr>
<td>sql_numiowaits</td>
<td>int</td>
<td>Number of times an I/O operation had to wait</td>
</tr>
<tr>
<td>sql_avgiowaits</td>
<td>float</td>
<td>Average amount of time that the SQL statement had to wait</td>
</tr>
<tr>
<td>sql_totaliowaits</td>
<td>float</td>
<td>Total amount of time that the SQL statement had to wait for I/O. This excludes any asynchronous I/O.</td>
</tr>
<tr>
<td>sql_rowspersec</td>
<td>float</td>
<td>Average number of rows (per second) produced</td>
</tr>
<tr>
<td>sql_estcost</td>
<td>int</td>
<td>Cost associated with the SQL statement</td>
</tr>
<tr>
<td>sql_estrows</td>
<td>int</td>
<td>Estimated number of rows returned for the SQL statement as predicted by the optimizer</td>
</tr>
<tr>
<td>sql_actualrows</td>
<td>int</td>
<td>Number of rows returned for the SQL statement</td>
</tr>
<tr>
<td>sql_sqlerror</td>
<td>int</td>
<td>SQL error number</td>
</tr>
<tr>
<td>sql_isamerror</td>
<td>int</td>
<td>RSAM/ISAM error number</td>
</tr>
<tr>
<td>sql_isollevel</td>
<td>int</td>
<td>Isolation level of the SQL statement.</td>
</tr>
<tr>
<td>sql_sqlmemory</td>
<td>int</td>
<td>Number of bytes needed to execute the SQL statement</td>
</tr>
<tr>
<td>sql_numiterators</td>
<td>int</td>
<td>Number of iterators used by the statement</td>
</tr>
<tr>
<td>sql_database</td>
<td>char(128)</td>
<td>Database name</td>
</tr>
<tr>
<td>sql_numtables</td>
<td>int</td>
<td>Number of tables used in executing the SQL statement</td>
</tr>
<tr>
<td>sql_tablelist</td>
<td>char(4096)</td>
<td>List of table names directly referenced in the SQL statement. If the SQL statement fires triggers that execute statements against other tables, the other tables are not listed.</td>
</tr>
<tr>
<td>sql_statement</td>
<td>char(1600)</td>
<td>SQL statement that ran</td>
</tr>
</tbody>
</table>

### syssqltrace_info

The syssqltrace_info table describes information about the SQL profile trace system.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flags</td>
<td>integer</td>
<td>SQL trace flags</td>
</tr>
<tr>
<td>ntraces</td>
<td>integer</td>
<td>Number of items to trace</td>
</tr>
<tr>
<td>tracesize</td>
<td>integer</td>
<td>Size of the text to store for each SQL trace item</td>
</tr>
<tr>
<td>duration</td>
<td>integer</td>
<td>Trace buffer (in seconds)</td>
</tr>
<tr>
<td>sqlseen</td>
<td>int8</td>
<td>Number of SQL items traced since start or resizing</td>
</tr>
<tr>
<td>startime</td>
<td>int8</td>
<td>Time tracing was enabled</td>
</tr>
<tr>
<td>memoryused</td>
<td>int8</td>
<td>Number of bytes of memory used by SQL tracing</td>
</tr>
</tbody>
</table>

Chapter 2. The sysmaster Database  2-25
### syssqltrace_iter

The syssqltrace_iter table lists the SQL statement iterators.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sql_id</td>
<td>int8</td>
<td>SQL execution ID</td>
</tr>
<tr>
<td>sql_address</td>
<td>int8</td>
<td>Address of the SQL statement block</td>
</tr>
<tr>
<td>sql_itr_address</td>
<td>int8</td>
<td>Address of the iterator</td>
</tr>
<tr>
<td>sql_itr_id</td>
<td>int</td>
<td>Iterator ID</td>
</tr>
<tr>
<td>sql_itr_left</td>
<td>int</td>
<td>Iterator ID to the left</td>
</tr>
<tr>
<td>sql_itr_right</td>
<td>int</td>
<td>Iterator ID to the right</td>
</tr>
<tr>
<td>sql_itr_cost</td>
<td>int</td>
<td>Iterator cost</td>
</tr>
<tr>
<td>sql_itr_estrows</td>
<td>int</td>
<td>Iterator estimated rows</td>
</tr>
<tr>
<td>sql_itr_numrows</td>
<td>int</td>
<td>Iterator actual rows processed</td>
</tr>
<tr>
<td>sql_itr_type</td>
<td>int</td>
<td>Iterator type</td>
</tr>
<tr>
<td>sql_itr_misc</td>
<td>int</td>
<td>Iterator miscellaneous flags</td>
</tr>
<tr>
<td>sql_it_info</td>
<td>char(256)</td>
<td>Iterator miscellaneous flags displayed as text</td>
</tr>
</tbody>
</table>

### syssrcrss

The syssrcrss table provides RS secondary server related statistics at the primary server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>int8</td>
<td>RS secondary server control block address</td>
</tr>
<tr>
<td>server_name</td>
<td>char(128)</td>
<td>Database server name</td>
</tr>
<tr>
<td>server_status</td>
<td>char(20)</td>
<td>Quiescent, active, or inactive</td>
</tr>
<tr>
<td>connection_status</td>
<td>char(20)</td>
<td>Connected or disconnected</td>
</tr>
<tr>
<td>log_transmission_status</td>
<td>char(20)</td>
<td>Active or blocked</td>
</tr>
<tr>
<td>next_page_tosend_log_uniq</td>
<td>integer</td>
<td>Log unique ID of next page to send</td>
</tr>
<tr>
<td>next_page_tosend_log_page</td>
<td>integer</td>
<td>Page number of next page to send</td>
</tr>
<tr>
<td>seq_tosend</td>
<td>integer</td>
<td>Sequence ID of last buffer sent</td>
</tr>
<tr>
<td>last_seq_acked</td>
<td>integer</td>
<td>Sequence ID of last buffer acknowledged</td>
</tr>
</tbody>
</table>

### syssrcsds

The syssrcsds table provides SD secondary server related statistics at the primary server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>int8</td>
<td>SD secondary server control block address</td>
</tr>
<tr>
<td>source_server</td>
<td>char(128)</td>
<td>Primary database server name</td>
</tr>
<tr>
<td>connection_status</td>
<td>char(20)</td>
<td>&quot;Connected&quot; or &quot;Disconnected&quot;</td>
</tr>
<tr>
<td>last_received_log_uniq</td>
<td>integer</td>
<td>Unique log ID of last log page received</td>
</tr>
<tr>
<td>last_received_log_page</td>
<td>integer</td>
<td>Page number of last log page received</td>
</tr>
<tr>
<td>next_lpgtoread_log_uniq</td>
<td>integer</td>
<td>Unique log ID of next log page to read</td>
</tr>
<tr>
<td>next_lpgtoread_log_page</td>
<td>integer</td>
<td>Page number of next log page to read</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>last_acked_lsn_uniq</td>
<td>integer</td>
<td>Unique log ID of last LSN acknowledged</td>
</tr>
<tr>
<td>last_acked_lsn_pos</td>
<td>integer</td>
<td>Log position of last LSN acknowledged</td>
</tr>
<tr>
<td>last_seq_received</td>
<td>integer</td>
<td>Sequence ID of last buffer received</td>
</tr>
<tr>
<td>last_seq_acked</td>
<td>integer</td>
<td>Sequence ID of last buffer acknowledged</td>
</tr>
<tr>
<td>cur_pagingfile</td>
<td>char(640)</td>
<td>Current paging file name</td>
</tr>
<tr>
<td>cur_pagingfile_size</td>
<td>int8</td>
<td>Current paging file size</td>
</tr>
<tr>
<td>old_pagingfile</td>
<td>char(640)</td>
<td>Old paging file name</td>
</tr>
<tr>
<td>old_pagingfile_size</td>
<td>int8</td>
<td>Old paging file size</td>
</tr>
</tbody>
</table>

**systabnames**

The **systabnames** table describes each table that the database server manages.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnum</td>
<td>integer</td>
<td>Tablespace identifier</td>
</tr>
<tr>
<td>dbname</td>
<td>char(128)</td>
<td>Database name</td>
</tr>
<tr>
<td>owner</td>
<td>char(32)</td>
<td>User ID of owner</td>
</tr>
<tr>
<td>tablename</td>
<td>char(128)</td>
<td>Table name</td>
</tr>
<tr>
<td>collate</td>
<td>char(32)</td>
<td>Collation associated with a database that supports GLS</td>
</tr>
</tbody>
</table>

**systhreads**

The **systhreads** table describes information about waiting threads.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>wreason</td>
<td>integer</td>
<td>Reason the thread is waiting</td>
</tr>
<tr>
<td>wait_reason</td>
<td>char(18)</td>
<td>Reason the thread is waiting</td>
</tr>
</tbody>
</table>

**systrgrss**

The **systrgrss** table provides RS secondary server related statistics at the RS secondary server.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>int8</td>
<td>RS secondary server control block address</td>
</tr>
<tr>
<td>source_server</td>
<td>char(128)</td>
<td>Source server serving the RS secondary server</td>
</tr>
<tr>
<td>connection_status</td>
<td>char(20)</td>
<td>Connected or disconnected</td>
</tr>
<tr>
<td>last_received_log_uniq</td>
<td>integer</td>
<td>Unique log ID of last log page received</td>
</tr>
<tr>
<td>last_received_log_page</td>
<td>integer</td>
<td>Page number of last log page received</td>
</tr>
<tr>
<td>last_seq_received</td>
<td>integer</td>
<td>Sequence ID of last buffer received</td>
</tr>
<tr>
<td>last_seq_acked</td>
<td>integer</td>
<td>Sequence ID of last buffer acknowledged</td>
</tr>
</tbody>
</table>

**systrgsds**

The **systrgsds** table provides SD secondary server related statistics at the SD secondary server.
<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>int8</td>
<td>SD secondary server control block address</td>
</tr>
<tr>
<td>source_server</td>
<td>char(128)</td>
<td>Source server serving the SD secondary server</td>
</tr>
<tr>
<td>connection_status</td>
<td>char(20)</td>
<td>Connected or disconnected</td>
</tr>
<tr>
<td>last_received_log_uniq</td>
<td>integer</td>
<td>Unique log ID of last log page received</td>
</tr>
<tr>
<td>last_received_log_page</td>
<td>integer</td>
<td>Page number of last log page received</td>
</tr>
<tr>
<td>next_lptoread_log_uniq</td>
<td>integer</td>
<td>Unique log ID of next log page to read</td>
</tr>
<tr>
<td>next_lptoread_log_page</td>
<td>integer</td>
<td>Page number of next log page to read</td>
</tr>
<tr>
<td>last_acked_lsn_uniq</td>
<td>integer</td>
<td>Unique log ID of last LSN acknowledged</td>
</tr>
<tr>
<td>last_acked_lsn_pos</td>
<td>integer</td>
<td>Log position of last LSN acknowledged</td>
</tr>
<tr>
<td>last_seq_received</td>
<td>integer</td>
<td>Sequence ID of last buffer received</td>
</tr>
<tr>
<td>last_seq_acked</td>
<td>integer</td>
<td>Sequence ID of last buffer acknowledged</td>
</tr>
<tr>
<td>cur_pagingfile</td>
<td>char(640)</td>
<td>Current paging file name</td>
</tr>
<tr>
<td>cur_pagingfile_size</td>
<td>int8</td>
<td>Current paging file size</td>
</tr>
<tr>
<td>old_pagingfile</td>
<td>char(640)</td>
<td>Old paging file name</td>
</tr>
<tr>
<td>old_pagingfile_size</td>
<td>int8</td>
<td>Old paging file size</td>
</tr>
</tbody>
</table>

**sysvprof**

The `sysvprof` table lists user and system CPU time for each virtual processor.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vpid</td>
<td>integer</td>
<td>Virtual processor ID</td>
</tr>
<tr>
<td>class</td>
<td>char(50)</td>
<td>Type of virtual processor:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• cpu</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• adm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• lio</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• pio</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• aio</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• tli</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• soc</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• str</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• shm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• opt</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• msc</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• adt</td>
</tr>
<tr>
<td>usercpu</td>
<td>float</td>
<td>Number of microseconds of user time</td>
</tr>
<tr>
<td>syscpu</td>
<td>float</td>
<td>Number of microseconds of system time</td>
</tr>
</tbody>
</table>

**The SMI Tables Map**

![Figure 2-1](image) displays the columns in some of the SMI tables.
<table>
<thead>
<tr>
<th>sysadtdinfo</th>
<th>sysaudit</th>
<th>syschkio</th>
<th>syschunks</th>
<th>sysconfig</th>
<th>syssdatabases</th>
</tr>
</thead>
<tbody>
<tr>
<td>adttmode</td>
<td>username</td>
<td>chunknum</td>
<td>chknum</td>
<td>cf_id</td>
<td>name</td>
</tr>
<tr>
<td>adterr</td>
<td></td>
<td>reads</td>
<td>dbsnm</td>
<td>cf_name</td>
<td>partnum</td>
</tr>
<tr>
<td>adtsize</td>
<td>succ1</td>
<td>pagesread</td>
<td>nxchknum</td>
<td>cf_flags</td>
<td>owner</td>
</tr>
<tr>
<td>adtpath</td>
<td>succ2</td>
<td>writes</td>
<td>chksize</td>
<td>cf_originals</td>
<td>created</td>
</tr>
<tr>
<td>adtfile</td>
<td>succ3</td>
<td>pageswritten</td>
<td>offset</td>
<td>cf_effective</td>
<td></td>
</tr>
<tr>
<td></td>
<td>succ4</td>
<td>mreads</td>
<td>nfree</td>
<td>cf_default</td>
<td></td>
</tr>
<tr>
<td></td>
<td>succ5</td>
<td>mpagesread</td>
<td>Is_offline</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fail1</td>
<td></td>
<td>mwrites</td>
<td>is_recovering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fail2</td>
<td></td>
<td>mpageswritten</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>fail3</td>
<td></td>
<td></td>
<td>is_blobchunk</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fail4</td>
<td></td>
<td></td>
<td>is_sbchunk</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fail5</td>
<td></td>
<td></td>
<td>is_inconsistent</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>flags</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>fname</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mfname</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>moffset</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mis_offline</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mis_recovering</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mis_recovery</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mflags</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 2-1. Columns in the SMI tables (Part 1 of 4)*

<table>
<thead>
<tr>
<th>sysdblocale</th>
<th>sysdbspaces</th>
<th>syddfri</th>
<th>sysextents</th>
<th>sysextspaces</th>
<th>syslocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbs_dbsname</td>
<td>dbsnum</td>
<td>type</td>
<td>dbsname</td>
<td>id</td>
<td>dbsname</td>
</tr>
<tr>
<td>dbs_collate</td>
<td>name</td>
<td>state</td>
<td>tabname</td>
<td>name</td>
<td>tabname</td>
</tr>
<tr>
<td></td>
<td>owner</td>
<td>name</td>
<td>chunk</td>
<td>name</td>
<td>owner</td>
</tr>
<tr>
<td></td>
<td>fchunk</td>
<td>timeout</td>
<td>offset</td>
<td>flags</td>
<td>keynum</td>
</tr>
<tr>
<td></td>
<td>nchunks</td>
<td></td>
<td>size</td>
<td>keynum</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>lostfound</td>
<td></td>
<td>refcnt</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>locsize</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>location</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 2-1. Columns in the SMI tables (Part 2 of 4)*
Information from onstat in the SMI Tables

To obtain information provided by the onstat utility, you can use SQL to query appropriate SMI tables. The following table indicates which SMI tables to query to obtain the information provided by a given onstat option. For descriptions of the onstat options, see “Monitor the Database Server Status” on page 15-4.

<table>
<thead>
<tr>
<th>onstat Option</th>
<th>SMI Tables to Query</th>
<th>onstat Fields Not in SMI Tables</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>sysdbspaces</td>
<td>address</td>
</tr>
<tr>
<td></td>
<td>syschunks</td>
<td>bpages</td>
</tr>
<tr>
<td>onstat Option</td>
<td>SMI Tables to Query</td>
<td>onstat Fields <em>Not</em> in SMI Tables</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>-D</td>
<td>sysdbspaces</td>
<td></td>
</tr>
<tr>
<td></td>
<td>syschkio</td>
<td></td>
</tr>
<tr>
<td>-F</td>
<td>sysprofile</td>
<td>address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flusher</td>
</tr>
<tr>
<td></td>
<td></td>
<td>snoozer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>state</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data</td>
</tr>
<tr>
<td>-g ath</td>
<td>systhreads</td>
<td></td>
</tr>
<tr>
<td>-g dri</td>
<td>sysdri</td>
<td>Last DR CKPT (id/pg)</td>
</tr>
<tr>
<td>-g glo</td>
<td>sysvpprof</td>
<td>Listing of virtual processors by class</td>
</tr>
<tr>
<td>-g ipl</td>
<td>sysipl</td>
<td></td>
</tr>
<tr>
<td>-g rss</td>
<td>syrsssllog</td>
<td></td>
</tr>
<tr>
<td></td>
<td>systrgrss</td>
<td></td>
</tr>
<tr>
<td></td>
<td>syssrcrss</td>
<td></td>
</tr>
<tr>
<td>-g his</td>
<td>syssqltracing</td>
<td></td>
</tr>
<tr>
<td>-g sds</td>
<td>syssrcsds</td>
<td></td>
</tr>
<tr>
<td></td>
<td>systrgds</td>
<td></td>
</tr>
<tr>
<td>-g smx</td>
<td>syssmx</td>
<td></td>
</tr>
<tr>
<td>-g smx ses</td>
<td>syssmxsres</td>
<td></td>
</tr>
<tr>
<td>-k</td>
<td>syslocks</td>
<td>address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>lklist</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tblsnum</td>
</tr>
<tr>
<td>-l</td>
<td>syslogs</td>
<td>All physical-log fields (except numpages and numwrits)</td>
</tr>
<tr>
<td></td>
<td>sysprofile</td>
<td>All logical-log buffer fields (except numrecs, numpages, and numwrits)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>address</td>
</tr>
<tr>
<td></td>
<td></td>
<td>begin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>% used</td>
</tr>
<tr>
<td>-p</td>
<td>sysprofile</td>
<td></td>
</tr>
<tr>
<td>-u</td>
<td>syssessions</td>
<td>address</td>
</tr>
<tr>
<td></td>
<td>syssesprof</td>
<td>wait</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nreads</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nwrites</td>
</tr>
</tbody>
</table>
Chapter 3. The sysadmin Database

In This Chapter

Chapter 3. The sysadmin Database ................................. 3-1
The sysadmin Database ................................................. 3-1
The PH_TASK Table ................................................... 3-1
The PH_RUN Table .................................................... 3-3
The PH_GROUP Table .................................................. 3-3
The PH_ALERT Table .................................................... 3-3
The PH_THRESHOLD Table .............................................. 3-5
The Results Table ...................................................... 3-5
The command_history Table .......................................... 3-5

In This Chapter

This chapter describes the sysadmin database and contains reference information about the tables in the database.

The sysadmin Database

The Scheduler is an administrative tool that enables the database server to execute database functions and procedures at predefined times or as determined internally by the server. The Scheduler is defined and driven by tasks, and the sysadmin database contains the six tables which contain and organize Scheduler task information. By default, only user informix is granted access to the sysadmin database; other users may be granted access to sysadmin. For detailed information about the Scheduler, see the IBM Informix Administrator’s Guide.

Because several important database server components use it, you should not drop or alter the sysadmin database. You can, however, move the sysadmin database from its default root dbspace location if the root dbspace does not have enough space for storing task properties and command history information. For instructions on moving the sysadmin database, see the procedure for using the RESET SYSADMIN SQL Administration API command in the IBM Informix Administrator’s Guide.

The PH_TASK Table

The PH_TASK table contains information about how and when each task will be executed.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tk_id</td>
<td>serial</td>
<td>Sequential task ID</td>
</tr>
<tr>
<td>tk_name</td>
<td>char(36)</td>
<td>Task name. A unique index on this column ensures that no two names are the same.</td>
</tr>
<tr>
<td>tk_description</td>
<td>lvarchar</td>
<td>Description about this task</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| tk_type       | char(18)       | Type of task:  
|               |                | • TASK: Executes a task which does not collect data  
|               |                | • SENSOR: A task which collects data  
|               |                | • STARTUP SENSOR: Runs only when the server starts  
|               |                | • STARTUP MONITOR: Runs only when the server starts                                                                                      |
| tk_sequence   | integer        | Current data collection number  
|               |                | System updated; do not modify                                                                                                             |
| tk_owner      | integer        | Owner’s thread ID  
|               |                | System updated; do not modify                                                                                                             |
| tk_result_table | varchar       | Result table name  
|               |                | Note: The tk_result_table column is used only by sensors and the content matches the table created in tk_create. When the tk_delete interval is exceeded, data is deleted from tk_result_table. |
| tk_create     | varchar        | The CREATE TABLE statement to execute  
|               |                | Note: The tk_create column is used only by sensors, and as necessary, is created to contain any data a sensor might store.               |
| tk_execute    | varchar        | The SQL object to execute                                                                                                                  |
| tk_delete     | interval day(2) to second | Deletes data older than this interval                                                                                |
| tk_start_time | datetime hour to second | Starting time of this task                                                                                                             |
| tk_stop_time  | datetime hour to second | Time of day this task should stop running.                                                                                           |
| tk_frequency  | interval day(2) to second | How often this task runs                                                                                                               |
| tk_next_execution | datetime year to second | Next time this task should be executed                                                                                               |
| tk_attributes | integer        | Flags  
|               |                | System updated; do not modify                                                                                                             |
| tk_group      | varchar(128)   | Group Name references  
|               |                | ph_group(group_name)                                                                                                                     |
| tk_exec_num   | integer        | Number of times to execute this task  
|               |                | System updated; do not modify                                                                                                             |
| tk_exec_time  | integer        | Total time spent executing this task  
|               |                | System updated; do not modify                                                                                                             |
| tk_enabled    | boolean        | Whether or not the task is enabled  
|               |                | If the value of tk_enabled equals FALSE, the task is not scheduled for execution                                                          |
| tk_priority   | integer        | Job priority, on a scale of 0- 5. If there are several jobs to execute simultaneously, the job with the highest priority executes first. The default is 0. |
The **PH_RUN** Table

The **PH_RUN** table contains information about how and when each Scheduler task ran.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>run_id</td>
<td>serial</td>
<td>Sequential ID generated during execution</td>
</tr>
<tr>
<td>run_task_id</td>
<td>integer</td>
<td>ID of the Scheduler task executed out of the PH_TASK table</td>
</tr>
<tr>
<td>run_task_seq</td>
<td>integer</td>
<td>Data Collector sequence number</td>
</tr>
<tr>
<td>run_recode</td>
<td>integer</td>
<td>Return code or SQLcode from the UDR or SQL statement</td>
</tr>
<tr>
<td>run_time</td>
<td>datetime</td>
<td>Time this Scheduler task was executed</td>
</tr>
<tr>
<td>run_duration</td>
<td>float</td>
<td>Time it took to execute this job (in seconds)</td>
</tr>
<tr>
<td>run_ztime</td>
<td>integer</td>
<td>Time <strong>onstat</strong> -z was last run</td>
</tr>
<tr>
<td>run_btime</td>
<td>integer</td>
<td>Time when server started</td>
</tr>
<tr>
<td>run_mttime</td>
<td>integer</td>
<td>Time the task was executed</td>
</tr>
</tbody>
</table>

The **PH_GROUP** Table

The **PH_GROUP** table contains information about the Scheduler group names.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>group_id</td>
<td>serial</td>
<td>Group ID</td>
</tr>
<tr>
<td>group_name</td>
<td>varchar(128)</td>
<td>Unique name of the group</td>
</tr>
<tr>
<td>group_description</td>
<td>lvarchar</td>
<td>Description of the group</td>
</tr>
</tbody>
</table>

The **PH_ALERT** Table

The **PH_ALERT** table contains information for the Scheduler about error, warning, or informational messages.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>serial</td>
<td>Alert ID</td>
</tr>
<tr>
<td>alert_run_id</td>
<td>integer</td>
<td>Invocation of a Scheduler task that created the alert</td>
</tr>
<tr>
<td>alert_task_seq</td>
<td>integer</td>
<td>Identifies which invocation of a Scheduler task created the alert</td>
</tr>
<tr>
<td>alert_type</td>
<td>char(8)</td>
<td>Informational, warning, or error</td>
</tr>
<tr>
<td>alert_color</td>
<td>char(15)</td>
<td>Green, yellow, or red. For more information about the alerts, see Table x below.</td>
</tr>
<tr>
<td>alert_time</td>
<td>datetime</td>
<td>Year to second Time the alert was generated</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>alert_state</td>
<td>char(15)</td>
<td>Indicates which state the object is in currently:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>NEW</strong> The alert was newly added and no other action has occurred on this alert.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>IGNORED</strong> The alert was acknowledged by the DBA and no action was taken.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>ACKNOWLEDGED</strong> The alert has been acknowledged by the DBA.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>ADDRESSED</strong> The alert has been addressed by the DBA.</td>
</tr>
<tr>
<td>alert_state_changed</td>
<td>datetime year to second</td>
<td>The last time the state was changed</td>
</tr>
<tr>
<td>alert_object_type</td>
<td>char(15)</td>
<td>The type of object:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SERVER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DATABASE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• TABLE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• INDEX</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• DBSPACE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• CHUNK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• USER</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SQL_STATEMENT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• MISC</td>
</tr>
<tr>
<td>alert_object_name</td>
<td>varchar(255)</td>
<td>The name of the object described above</td>
</tr>
<tr>
<td>alert_message</td>
<td>lvarchar</td>
<td>Message</td>
</tr>
<tr>
<td>alert_action</td>
<td>lvarchar</td>
<td>Corrective Action. This is an SQL script which can be executed by the user or tool or it will be NULL if no action is available. This script must comply with all multi-statement prepare rules.</td>
</tr>
<tr>
<td>alert_action_dbs</td>
<td>lvarchar(256)</td>
<td>Name of the database to use when executing the alert_action</td>
</tr>
</tbody>
</table>

This table defines the alert colors for the three different types of messages.

<table>
<thead>
<tr>
<th></th>
<th>Green</th>
<th>Yellow</th>
<th>Red</th>
</tr>
</thead>
<tbody>
<tr>
<td>Informative</td>
<td>A status message indicating a component's operation status</td>
<td>An important status message</td>
<td>A status message that requires action.</td>
</tr>
<tr>
<td>Warning</td>
<td>A warning from the database that was automatically addressed</td>
<td>A future event that needs to be addressed</td>
<td>A predicted failure is imminent. Action is necessary now.</td>
</tr>
</tbody>
</table>
### The PH_THRESHOLD Table

The **PH_THRESHOLD** table contains information about thresholds for Scheduler tasks.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>integer</td>
<td>Alert ID</td>
</tr>
<tr>
<td>task_name</td>
<td>varchar</td>
<td>Scheduler ask name associated with the threshold</td>
</tr>
<tr>
<td>Name</td>
<td>char</td>
<td>Name of the threshold</td>
</tr>
<tr>
<td>Value</td>
<td>lvarchar</td>
<td>Value of the threshold</td>
</tr>
<tr>
<td>Value_Type</td>
<td>char</td>
<td>The data type of the value column:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• STRING</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NUMERIC</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• NUMERIC, MAX, MIN</td>
</tr>
<tr>
<td>Description</td>
<td>lvarchar</td>
<td>Description of the threshold</td>
</tr>
</tbody>
</table>

### The Results Table

The Results table contains historical data about Scheduler task execution.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>integer</td>
<td>Required Column and Name This column links to the <strong>PH_RUN</strong> table.</td>
</tr>
<tr>
<td>USER COLUMNS</td>
<td>any</td>
<td>User Column</td>
</tr>
</tbody>
</table>

### The command_history Table

The **command_history** table contains a list of all commands that the Administration API ran. The table also shows the results of the commands. This table, which is in the **sysadmin** database, is a RAW (nonlogged) table.

The **command_history** table shows if an administrative task was executed through an **admin()** or **task()** function and displays information about the user who executed the command, the time the command was executed, the command, and the message returned when the database server completed running the command.

**Table 3-1. Example Showing command_history Table Information**

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmd_number</td>
<td>serial</td>
<td>Unique ID for each row</td>
</tr>
<tr>
<td>cmd_exec_time</td>
<td>datetime year-to-second</td>
<td>Time the command started</td>
</tr>
</tbody>
</table>
Table 3-1. Example Showing command_history Table Information (continued)

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmd_user</td>
<td>varchar</td>
<td>User who executed the command</td>
</tr>
<tr>
<td>cmd_hostname</td>
<td>varchar</td>
<td>Name of the host computer from which the command was executed</td>
</tr>
<tr>
<td>cmd_executed</td>
<td>varchar</td>
<td>The command that was executed</td>
</tr>
<tr>
<td>cmd_ret_status</td>
<td>integer</td>
<td>Return code</td>
</tr>
<tr>
<td>cmd_ret_msg</td>
<td>lvarchar</td>
<td>Return message</td>
</tr>
</tbody>
</table>

The following table shows sample commands and the associated results in a command_history table.

Table 3-2. Example of some Information in a command_history Table

<table>
<thead>
<tr>
<th>Command Executed</th>
<th>Sample Returned Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td>set sql tracing on</td>
<td>SQL tracing on with 1000 buffers of 2024 bytes.</td>
</tr>
<tr>
<td>create dbspace</td>
<td>Space ‘space12’ added.</td>
</tr>
<tr>
<td>checkpoint</td>
<td>Checkpoint completed.</td>
</tr>
<tr>
<td>add log</td>
<td>Added 3 logical logs to dbspace logdbs.</td>
</tr>
</tbody>
</table>

To display the command history, run this SQL statement:

SELECT * from command_history

Task in the command_history table are automatically removed after a fixed period of time. You can modify this time period by changing information in the COMMAND HISTORY RETENTION row in the ph_threshold table. The COMMAND HISTORY RETENTION parameter sets the length of time rows should remain in the command_history table.

You can use SQL commands like delete or truncate table to manually remove data from this table.
Chapter 4. Disk Structures and Storage

In This Chapter. .......................... 4-2
Db space Structure and Storage. .......................... 4-2
  Structure of the Root Db space .......................... 4-2
Reserved Pages .................................. 4-3
Structure of a Regular Db space .......................... 4-3
  Structure of an Additional Db space Chunk ............... 4-3
  Structure of a Mirror Chunk .......................... 4-4
Structure of the Chunk Free-List Page ....................... 4-4
Structure of the Tblspace Tblspace .......................... 4-5
  Tblspace Tblspace Entries .......................... 4-5
  Tblspace Numbers .................................. 4-6
  Tblspace Number Elements .................................. 4-7
  Tblspace Tblspace Size .................................. 4-7
  Tblspace Tblspace Bitmap Page .......................... 4-7
Structure of the Database Tblspace .......................... 4-7
  Database Tblspace Number .................................. 4-7
  Database Tblspace Entries .................................. 4-7
Structure and Allocation of an Extent .......................... 4-8
  Extent Structure. .................................. 4-8
  Next-Extent Allocation .................................. 4-11
Structure and Storage of a Db space Page .......................... 4-12
  Rows in Nonfragmented Tables .......................... 4-13
  Rows in Fragmented Tables .................................. 4-14
  Recommendations on Use of Rowid .......................... 4-14
  Data-Row Format and Storage .................................. 4-14
Structure of Fragmented Tables .......................... 4-15
  Attached Indexes .................................. 4-16
  Detached Indexes .................................. 4-16
  Structure of B-Tree Index Pages .......................... 4-16
  Definition of B-Tree Terms .................................. 4-16
  Logical Storage of Indexes .................................. 4-17
  Functional Indexes .................................. 4-20
Structure of R-Tree Index Pages .......................... 4-21
Storage of Simple Large Objects .......................... 4-21
  Structure of a Blob space .................................. 4-21
  Structure of a Db space Blobpage .......................... 4-21
Simple-Large-Object Storage and the Descriptor .......................... 4-22
  Creation of Simple Large Objects .......................... 4-22
  Deletion or Insertion of Simple Large Objects .......................... 4-22
  Size Limits for Simple Large Objects .......................... 4-22
Blob space Page Types .......................... 4-22
  Blob space Free-Map Page .................................. 4-22
  Blob space Bitmap Page .................................. 4-23
  Blobpage .................................. 4-23
  Structure of a Blob space Blobpage .......................... 4-23
Blob space Structure .................................. 4-23
  Structure of the Metadata Area .................................. 4-24
Blob page Structure .................................. 4-25
Multiple Chunk Blob space .................................. 4-25
Time Stamps .................................. 4-26
Database and Table Creation: What Happens on Disk. .......................... 4-26
  Database Creation .................................. 4-26
  Disk-Space Allocation for System Catalog Tables .......................... 4-26
  Tracking of System Catalog Tables .................................. 4-27
  Table Creation .................................. 4-27
In This Chapter

The database server achieves its high performance by managing its own I/O. The database server manages storage, search, and retrieval. As the database server stores data, it creates the structures it needs to search for and retrieve the data later. The database server disk structures also store and track control information needed to manage logging and backups. Database server structures contain all the information needed to ensure data consistency, both physical and logical.

Before you read this chapter, familiarize yourself with the disk-space terms and definitions in the chapter on where data is stored in the *IBM Informix Administrator’s Guide*.

This chapter discusses the following topics related to disk data structures:

- Dbspace structure and storage
- Storage of simple large objects
- Sbspace structure
- Time stamps
- Database and table creation: what happens on disk

**Dbspace Structure and Storage**

This section explores the disk structures and storage techniques that the database server uses to store data in a dbspace.

**Structure of the Root Dbspace**

The ROOTNAME, ROOTOFFSET, ROOTPATH, and ROOTSIZE configuration parameters specify the size and location of the initial chunk of the root dbspace. If the root dbspace is mirrored, the MIRROROFFSET and MIRRORPATH configuration parameters specify the mirror-chunk location. For more information about these parameters, see Chapter 1, “Configuration Parameters,” on page 1-1.

As part of disk-space initialization, the database server initializes the following structures in the initial chunk of the root dbspace:

- Twelve reserved pages
- The first chunk free-list page
- The tblspace tblspace
- The database tblspace
- The physical log
- The logical-log files
- **oncheck -pe**

For more information, see oncheck -ce, -pe: Check the chunk-free list on page 7-10.
Reserved Pages
The first 12 pages of the initial chunk of the root dbspace are reserved pages. Each reserved page contains specific control and tracking information used by the database server.

To obtain a listing of the contents of your reserved pages, execute the command **oncheck -pr**. To also list information about the physical-log and logical-log pages, including the active physical-log pages, execute **oncheck -pR**.

The following example shows **oncheck -pr** output for interval checkpoints:

```
<table>
<thead>
<tr>
<th>Time of checkpoint</th>
<th>10/25/2005 17:05:20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checkpoint Interval</td>
<td>1234</td>
</tr>
</tbody>
</table>
```

Figure 4-1. oncheck -pr Output

For examples of PAGE_CONFIG reserved page and logical-log file **oncheck -pr** output, see IBM Informix Administrator's Guide.

Structure of a Regular Dbspace
After disk-space initialization, you can add new dbspaces. When you create a dbspace, you assign at least one chunk (either raw or cooked disk space) to the **dbspace**. This chunk is referred to as the initial chunk of the dbspace. **Figure 4-2 on page 4-3** illustrates the structure of the initial chunk of a regular (nonroot) dbspace.

When the dbspace is first created, it contains the following structures:
- Two reserved pages
- The first chunk free-list page in the chunk
- The tblspace **tblspace** for this dbspace
- Unused pages

![Initial Chunk of Regular Dbspace](image)

Figure 4-2. Initial Chunk of Regular Dbspace

Structure of an Additional Dbspace Chunk
You can create a dbspace that contains more than one chunk. The initial chunk in a dbspace contains the tblspace **tblspace** for the dbspace. Additional chunks do not. When an additional chunk is first created, it contains the following structures:
- Two reserved pages
- The first chunk free-list page
- Unused pages
Figure 4-3 illustrates the structure of all additional chunks in a dbspace. (The structure also applies to additional chunks in the root dbspace.)

Structure of a Mirror Chunk
Each mirror chunk must be the same size as its primary chunk. When a mirror chunk is created, the database server writes the contents of the primary chunk to the mirror chunk immediately.

The mirror chunk contains the same control structures as the primary chunk. Mirrors of blobspace, sbspace, or dbspace chunks contain the same physical contents as their primary counterpart after the database server brings them online.

Figure 4-4 illustrates the mirror-chunk structure as it appears after the chunk is created.

Structure of the Chunk Free-List Page
In every chunk, the page that follows the last reserved page is the first of one or more chunk free-list pages that tracks available space in the chunk. For a non-root chunk, the initial length of the free space is equal to the size of the chunk minus three pages. If an additional chunk free-list page is needed to accommodate new entries, a new chunk free-list page is created in one of the free pages in the chunk.

Figure 4-5 illustrates the location of the free-list page.
Use `oncheck -pe` to obtain the physical layout of pages in the chunk. For more information, see "oncheck -ce, -pe: Check the chunk-free list" on page 7-10.

Structure of the Tblspace

Each dbspace contains a tblspace called the `tblspace tblspace` that describes all tblspaces in the dbspace. When the database server creates a tblspace, it places an entry in the tblspace `tblspace` that describes the characteristics of the newly created tblspace. You cannot drop or move a chunk containing a tblspace `tblspace`.

A dbspace can have a maximum number of $2^{**20}$ tblspaces.

The default size of the first and next extents depends on whether the dbspace is the root dbspace or not, as shown in the following table.

Table 4-1.

<table>
<thead>
<tr>
<th>Type of dbspace</th>
<th>Default Size of First Extent</th>
<th>Default Size of Next Extents</th>
</tr>
</thead>
</table>
| Root            | • 500 KB for a 2 kilobyte page system  
|                 | • 1000 KB for a 4 kilobyte page system | • 100 KB for a 2 kilobyte page system  
|                 |                              | • 200 KB for a 4 kilobyte page system |
| Non-root        | • 100 KB for a 2 kilobyte page system  
|                 | • 200 KB for a 4 kilobyte page system | • 100 KB for a 2 kilobyte page system  
|                 |                              | • 200 KB for a 4 kilobyte page system |

You can specify a non-default size for the first and next extents for a tblspace `tblspace` in the following ways:

- For the root dbspace, set the TBLTBLFIRST and TBLTBLNEXT configuration parameters.
- For non-root dbspaces, use the `onspaces` utility `-ef` and `-en` options when you create a dbspace.

Tblspace Tblspace Entries

To display information on the tblspace, use the `oncheck -pt` command. For more information, see "oncheck -pt and -pT: Display tblspaces for a table or fragment" on page 7-18.

Component | Description
-----------|--------------
Page header | 24 bytes, standard page-header information
Page-ending time stamp | 4 bytes
Tblspace header | 68 bytes, general tblspace information
Component | Description
--- | ---
Column information | Each special column in the table is tracked with an 12-byte entry. (A special column is defined as a VARCHAR, BYTE, or TEXT data type.)
Tblspace name | 80 bytes, database.owner.tablename
Index information | Each index on the table contains a 20-byte header that contains general information about the index, followed by a 4-byte entry for each column component of the index
Extent information | Each extent allocated to this tblspace is tracked with a 12-byte entry

### Tblspace Numbers

Each tblspace that is described in the tblspace tblspace receives a tblspace number. This tblspace number is the same value that is stored as the partnum field in the systables system catalog table and as the part field in the sysfragments system catalog table.

The following SQL query retrieves the partnum for every table in the database (these can be located in several different dbspaces) and displays it with the table name and the hexadecimal representation of partnum:

```sql
SELECT tabname, tabid, partnum, HEX(partnum) hex_tblspace_name FROM systables
```

If the output includes a row with a table name but a partnum of 0, this table consists of two or more table fragments, each located in its own tblspace. For example, Figure 4-6 shows a table called account that has partnum 0.

<table>
<thead>
<tr>
<th>tabname</th>
<th>tabid</th>
<th>partnum</th>
<th>hex_tblspace_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>sysfragments</td>
<td>25</td>
<td>1048611</td>
<td>0x00100023</td>
</tr>
<tr>
<td>branch</td>
<td>100</td>
<td>1048612</td>
<td>0x00100024</td>
</tr>
<tr>
<td>teller</td>
<td>101</td>
<td>1048613</td>
<td>0x00100025</td>
</tr>
<tr>
<td>account</td>
<td>102</td>
<td>0</td>
<td>0x00000000</td>
</tr>
<tr>
<td>history</td>
<td>103</td>
<td>1048615</td>
<td>0x00100027</td>
</tr>
<tr>
<td>results</td>
<td>104</td>
<td>1048616</td>
<td>0x00100028</td>
</tr>
</tbody>
</table>

*Figure 4-6. Output from systables Query with partnum Values*

To obtain the actual tblspace numbers for the fragments that make up the table, you must query the sysfragments table for the same database. Figure 4-7 shows that the account table from Figure 4-6 has three table fragments and three index fragments.

<table>
<thead>
<tr>
<th>tabid</th>
<th>fragtype</th>
<th>partn</th>
<th>hex_tblspace_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>102</td>
<td>T</td>
<td>1048614</td>
<td>0x00100026</td>
</tr>
<tr>
<td>102</td>
<td>T</td>
<td>2097154</td>
<td>0x00200002</td>
</tr>
<tr>
<td>102</td>
<td>T</td>
<td>3145730</td>
<td>0x00300002</td>
</tr>
<tr>
<td>102</td>
<td>I</td>
<td>1048617</td>
<td>0x00100029</td>
</tr>
<tr>
<td>102</td>
<td>I</td>
<td>2097155</td>
<td>0x00200003</td>
</tr>
<tr>
<td>102</td>
<td>I</td>
<td>3145731</td>
<td>0x00300003</td>
</tr>
</tbody>
</table>

*Figure 4-7. Output from sysfragments Table with partn Values*
**Tblspace Number Elements**

The first page in a tblspace is logical page 0. (Physical page numbers refer to the address of the page in the chunk.) The root space tblspace tblspace is always contained in the first dbspace and on logical page 1 within the tblspace tblspace. (The bitmap page is page 0.)

**Tblspace Tblspace Size**

These tblspace tblspace pages are allocated as an extent when the dbspace is initialized. If the database server attempts to create a table, but the tblspace tblspace is full, the database server allocates a next extent to the tblspace.

When a table is removed from the dbspace, its corresponding entry in the tblspace tblspace is deleted.

**Tblspace Tblspace Bitmap Page**

The first page of the tblspace tblspace, like the first page of any initial extent, is a bitmap that describes the page fullness of the following pages. Each page that follows has an entry on the bitmap page. If needed, additional bitmap pages are located throughout the contiguous space allocated for the tblspace, arranged so that each bitmap describes only the pages that follow it, until the next bitmap or the end of the dbspace. Bitmap pages fall at distinct intervals within tblspaces pages. Each bitmap page describes a fixed number of pages that follow it.

**Structure of the Database Tblspace**

The database tblspace appears only in the initial chunk of the root dbspace. The database tblspace contains one entry for each database managed by the database server. Figure 4-8 illustrates the location of the database tblspace.

![Diagram](image)

*Figure 4-8. Database Tblspace Location in Initial Chunk of Root Dbspace*

**Database Tblspace Number**

The tblspace number of the database tblspace is always 0x100002. This tblspace number appears in an onstat -t listing if the database tblspace is active.

**Database Tblspace Entries**

Each database tblspace entry includes the following five components:

- Database name
- Database owner
- Date and time that the database was created
• The tblspace number of the systables system catalog table for this database
• Flags that indicate logging mode

The database tblspace includes a unique index on the database name to ensure that every database is uniquely named. For any database, the systables table describes each permanent table in the database. Therefore, the database tblspace only points to the detailed database information located elsewhere.

When the root dbspace is initialized, the database tblspace first extent is allocated. The initial-extent size and the next-extent size for the database tblspace are four pages. You cannot modify these values.

Structure and Allocation of an Extent
This section covers the following topics:
• Extent structure
• Next-extent allocation

Extent Structure
An extent is a collection of contiguous pages within a dbspace. Every permanent database table has two extent sizes associated with it. The initial-extent size is the number of kilobytes allocated to the table when it is first created. The next-extent size is the number of kilobytes allocated to the table when the initial extent, and every extent thereafter, becomes full.

Blobspaces do not use extents.

For specific instructions on how to specify and calculate the size of an extent, see your IBM Informix Performance Guide.

Extent Size: The minimum size of an extent is four pages. The default size of an extent is eight pages. The maximum size of an extent is 2**31 pages, equivalent to the maximum chunk size. If the chunk is smaller than the maximum size, the maximum extent size depends on the contiguous space available in the chunk.

Tblspaces that hold index fragments follow different rules for extent size. The
database server bases the extent size for these tblspaces on the extent size for the
the corresponding table fragment. The database server uses the ratio of the row size to
index key size to assign an appropriate extent size for the index tblspace (see the
sections on estimating index page size and fragmenting table indexes in the IBM
Informix Performance Guide).

Page Types Within a Table Extent: Within the extent, individual pages contain
different types of data. Extent pages for a table can be separated into the following
categories:
• Data pages
  Data pages contain the data rows for the table.
• Bitmap pages
  Bitmap pages contain control information that monitors the fullness of every
  page in the extent.
• Blobpages
  Blobpages contain TEXT and BYTE data that is stored with the data rows in the
dbspace. TEXT and BYTE data that resides in a blobspace is stored in blobpages,
a structure that is completely different than the structure of a dbspace blobpage.
• Free pages
  Free pages are pages in the extent that are allocated for tblspace use, but whose function has not yet been defined. Free pages can be used to store any kind of information: data, including TEXT or BYTE data types; index; or bitmap.

Figure 4-9 on page 4-9 illustrates the possible structure of a nonfragmented table with an initial-extent size of 8 pages and a next-extent size of 16 pages.

![Extent Structure of a Table](image)

**Page Types Within an Index Extent:** The database server stores index pages into different tblspaces than the table with which it is associated. Within the extent, individual index pages contain different types of data. Index pages can be separated into the following categories:

• Index pages (root, branch, and leaf pages)
  Index pages contain the index information for the table.

• Bitmap pages
  Bitmap pages contain control information that monitors the fullness of every page in the extent.
- Free pages

Free pages are pages in the extent that are allocated for tblspace use, but whose function has not yet been defined. Free pages can be used to store any kind of information: data, index, TEXT or BYTE data, or bitmap.

All indexes are detached unless you explicitly specify attached indexes.

**Important:** An extent that is allocated for a table fragment does not contain index pages. Index pages for a fragmented table always reside in a separate tblspace. For more information, see fragmenting table indexes in the chapter on table fragmentation and PDQ in the *IBM Informix Administrator’s Guide*.

Figure 4-10 on page 4-10 illustrates the extent structure of an index.
Next-Extent Allocation
After the initial extent fills, the database server attempts to allocate another extent of contiguous disk space. The procedure that the database server follows is referred to as next-extent allocation.

Extents for a tblspace are tracked as one component of the tblspace tblspace information for the table. The maximum number of extents allocated for any tblspace is application and machine dependent because it varies with the amount of space available on the tblspace tblspace entry.

Next-Extent Size: The number of kilobytes that the database server allocates for a next extent is, in general, equal to the size of a next extent, as specified in the SQL statement CREATE TABLE. However, the actual size of the next-extent allocation might deviate from the specified size because the allocation procedure takes into account the following three factors:
- Number of existing extents for this tblspace
- Availability of contiguous space in the chunk and dbspace
- Location of existing tblspace extents

The effect of each of these factors on next-extent allocation is explained in the paragraphs that follow and in Figure 4-11 on page 4-12.

Extent Size Doubling: If a permanent table or user-defined temporary table already has 16 extents allocated, the database server automatically doubles the size for subsequent allocations. This doubling occurs every 16 extents. For example, if you create a table with NEXT SIZE equal to 20 kilobytes, the database server allocates the first 16 extents at a size of 20 kilobytes each. The database server allocates extents 17 to 32 at 40 kilobytes each, extents 33 to 48 at 80 kilobytes each, and so on.

The extent size doubling is allowed only if the total number of pages allocated so far is at least 16 times the current next extent size. This is a precautionary measure to limit the exponential doubling of next extent size, if the system has many small holes of pages less than the next extent size, thereby creating a greater number of small extents.

For system-created temporary tables, the next-extent size begins to double after 4 extents have been added.

Lack of Contiguous Space: If the database server cannot find available contiguous space in the first chunk equal to the size specified for the next extent, it extends the search to the next chunk in the dbspace. Extents are not allowed to span chunks.

If the database server cannot find adequate contiguous space anywhere in the dbspace, it allocates to the table the largest available amount of contiguous space. (The minimum allocation is four pages. The default value is eight pages.) No error message is returned if an allocation is possible, even when the amount of space allocated is less than the requested amount.

Merge of Extents for the Same Table: If the disk space allocated for a next extent is physically contiguous with disk space already allocated to the same table, the database server allocates the disk space but does not consider the new allocation as a separate extent. Instead, the database server extends the size of the existing contiguous extent. Thereafter, all disk-space reports reflect the allocation as an
extension of the existing extent. That is, the number of extents reported is always the number of physically distinct extents, not the number of times a next extent has been allocated plus one (the initial extent). Figure 4-11 illustrates extent-allocation strategies.

After disk space is allocated to a tblspace as part of an extent, the space remains dedicated to that tblspace even if the data contained in it is deleted. For alternative methods of reclaiming this empty disk space, see your IBM Informix Performance Guide.

Structure and Storage of a Dbspace Page
The basic unit of database server I/O is a page. Page size might vary among computers.

In Dynamic Server, the page size depends on the operating system.
Rows in Nonfragmented Tables

The database server can store rows that are longer than a page. The database server also supports the VARCHAR data type, which results in rows of varying length. As a result, rows do not conform to a single format.

Rows within a table are not necessarily the same length if the table contains one or more columns of type VARCHAR. In addition, the length of a row in such a table might change when an end user modifies data contained in the VARCHAR column.

The length of a row can be greater than a page.

TEXT and BYTE data is not stored within the data row. Instead, the data row contains a 56-byte descriptor that points to the location of the data. The descriptor can point to a dbspace page.

The descriptor can point to a blobspace blobpage. If you are using the Optical Subsystem, the descriptor can also point to an optical-storage subsystem.

For instructions about how to estimate the length of fixed-length and variable-length data rows, see your IBM Informix Performance Guide.

Definition of Rowid: Informix uses two different types of rowids to identify data in tables:
- **Serial rowid**
  These rowids are fields in a table and are assigned to tables created with the WITH ROWID option.
- **Internal rowid**
  The database server identifies each data row in a table with a unique internal rowid. This rowid identifies the location of the row within the dbspace.

To obtain the internal rowids for a table, use the `oncheck -pD` option. For more information, see "oncheck -cd and -cD: Check pages" on page 7-9.

In a nonfragmented table, the term rowid refers to a unique 4-byte integer that defines the physical location of the row in the table. The page that contains the first byte of the data row is the page that is specified by the rowid. This page is called the data row home page.

Fragmented tables can also have rowids, but they are implemented in a different way. For more information on this topic, see "Rows in Fragmented Tables" on page 4-14.

Use of Rowids: Every data row in a nonfragmented table is uniquely identified by an unchanging rowid. When you create an index for a nonfragmented table, the rowid is stored in the index pages associated with the table to which the data row belongs. When the database server requires a data row, it searches the index to find the key value and uses the corresponding rowid to locate the requested row. If the table is not indexed, the database server might sequentially read all the rows in the table.

Eventually, a row might outgrow its original storage location. If this occurs, a forward pointer to the new location of the data row is left at the position defined by the rowid. The forward pointer is itself a rowid that defines the page and the location on the page where the data row is now stored.
**Rows in Fragmented Tables**

Unlike rows in a nonfragmented table, the database server does not assign a rowid to rows in fragmented tables. If you want to access data by rowid, you must explicitly create a rowid column as described in your IBM Informix Performance Guide. If user applications attempt to reference a rowid in a fragmented table that does not contain a rowid that you explicitly created, the database server returns an appropriate error code to the application.

**Access to Data in Fragmented Tables with Rowid:** From the viewpoint of an application, the functionality of a rowid column in a fragmented table is identical to the rowid of a nonfragmented table. However, unlike the rowid of a nonfragmented table, the database server uses an index to map the rowid to a physical location.

When the database server accesses a row in a fragmented table using the rowid column, it uses this index to look up the physical address of the row before it attempts to access the row. For a nonfragmented table, the database server uses direct physical access without an index lookup. As a consequence, accessing a row in a fragmented table using rowid takes slightly longer than accessing a row using rowid in a nonfragmented table. You should also expect a small performance impact on the processing of inserts and deletes due to the cost of maintaining the rowid index for fragmented tables.

Primary-key access can lead to significantly improved performance in many situations, particularly when access is in parallel.

**Recommendations on Use of Rowid**

It is recommended that application developers use primary keys as a method of access rather than rowids. Because primary keys are defined in the ANSI specification of SQL, using them to access data makes your applications more portable.

For a complete description on how to define and use primary keys to access data, see the IBM Informix Guide to SQL: Reference and the IBM Informix Guide to SQL: Tutorial.

**Data-Row Format and Storage**

The variable length of a data row has the following consequences for row storage:

- A page might contain one or more whole rows.
- A page might contain portions of one or more rows.
- A page might contain a combination of whole rows and partial rows.
- An updated row might increase in size and become too long to return to its original storage location in a row.

The following paragraphs describe the guidelines that the database server follows during data storage.

**Storage of Row:** To minimize retrieval time, rows are not broken across page boundaries unnecessarily. Rows that are shorter than a page are always stored as whole rows. A page is considered full when the count of free bytes is less than the number of bytes needed to store a row of maximum size.

**Location of Rows:** When the database server receives a row that is longer than a page, the row is stored in as many whole pages as required. The database server then stores the trailing portion in less than a full page.
The page that contains the first byte of the row is the row home page. The number of the home page becomes the logical page number contained in the rowid. Each full page that follows the home page is referred to as a big-remainder page. If the trailing portion of the row is less than a full page, it is stored on a remainder page.

After the database server creates a remainder page to accommodate a long row, it can use the remaining space in this page to store other rows. 

Figure 4-12 illustrates the concepts of home page, big-remainder page, and remainder page.

![Figure 4-12. Remainder Pages](image)

**Page Compression:** Over time, the free space on a page can become fragmented. When the database server attempts to store data, it first checks row length against the number of free bytes on a page to determine if the row fits. If adequate space is available, the database server checks if the page contains adequate contiguous free space to hold the row (or row portion). If the free space is not contiguous, the database server calls for page compression.

**Structure of Fragmented Tables**

Although table fragmentation is transparent to applications, as database server administrator you should be aware of how the database server allocates disk space for table fragments and how the database server identifies rows in those fragments.

Each table fragment has its own tblspace with a unique tblspace_id or fragment_id. Figure 4-13 shows the disk allocation for a fragmented table that resides in different partitions of the same dbspace.
Attached Indexes
With an attached index, the index and data are fragmented in the same way. You can decide whether to store the index pages with the corresponding data pages in the same dbspace or store them in separate dbspaces. For information on choosing a fragmentation strategy, see the IBM Informix Performance Guide.

Detached Indexes
For detached indexes, the table fragment and index fragment are stored in tblspaces in separate dbspaces.

Structure of B-Tree Index Pages
This section provides general information about the structure of B-tree index pages. It is designed as an overview for the interested reader. For more information on B-tree indexes, see your IBM Informix Performance Guide.

Definition of B-Tree Terms
The database server uses a B-tree structure to organize index information. Figure 4-14 shows that a fully developed B-tree index is composed of the following three different types of index pages or nodes:

- **One root node**
  A root node contains node pointers to branch nodes.

- **Two or more branch nodes**
  A branch node contains pointers to leaf nodes or other branch nodes.

- **Many leaf nodes**
  A leaf node contains index items and horizontal pointers to other leaf nodes.

Each node serves a different function. The following sections describe each node and the role that it plays in indexing.
Index Items: The fundamental unit of an index is the index item. An index item contains a key value that represents the value of the indexed column for a particular row. An index item also contains rowid information that the database server uses to locate the row in a data page.

Nodes: A node is an index page that stores a group of index items. For the three types of nodes, see “Definition of B-Tree Terms” on page 4-16.

Logical Storage of Indexes
This section presents an overview of how the database server creates and fills an index.

Creation of Root and Leaf Nodes: When you create an index for an empty table, the database server allocates a single index page. This page represents the root node and remains empty until you insert data in the table.

At first, the root node functions in the same way as a leaf node. For each row that you insert into the table, the database server creates and inserts an index item in the root node. Figure 4-15 illustrates how a root node appears before it fills.

![Figure 4-14. Full B-Tree Structure](image)

When the root node becomes full of index items, the database server splits the root node by performing the following steps:

- Creates two leaf nodes
- Moves approximately half of the root-node entries to each of the newly created leaf nodes
- Puts pointers to leaf nodes in the root node
As you add new rows to a table, the database server adds index items to the leaf nodes. When a leaf node fills, the database server creates a new leaf node, moves part of the contents of the full index node to the new node, and adds a node pointer to the new leaf node in the root node.

For example, suppose that leaf node 3 in Figure 4-16 becomes full. When this situation occurs, the database server adds yet another leaf node. The database server moves part of the records from leaf node 3 to the new leaf node, as Figure 4-16 shows.

![Figure 4-16. Leaf Node 4 Created After Leaf Node 3 Fills](image)

**Creation of Branch Nodes:** Eventually, as you add rows to the table, the database server fills the root node with node pointers to all the existing leaf nodes. When the database server splits yet another leaf node, and the root node has no room for an additional node pointer, the following process occurs.

The database server splits the root node and divides its contents among two newly created branch nodes. As index items are added, more and more leaf nodes are split, causing the database server to add more branch nodes. Eventually, the root node fills with pointers to these branch nodes. When this situation occurs, the database server splits the root node again. The database server then creates yet another branch level between the root node and the lower branch level. This process results in a four-level tree, with one root node, two branch levels, and one leaf level. The B-tree structure can continue to grow in this way to a maximum of 20 levels.

Branch nodes can point either to other branch nodes below them (for large indexes of four levels or more) or to leaf nodes. In Figure 4-17 the branch node points to leaf nodes only. The first item in the left branch node contains the same key value as the largest item in the leftmost leaf node and a node pointer to it. The second item contains the largest item in the next leaf node and a node pointer to it. The third item in the branch node contains only a pointer to the next higher leaf node. Depending on the index growth, this third item can contain the actual key value in addition to the pointer at a later point during the lifespan of the index.
Duplicate Key Values: Duplicate key values occur when the value of an indexed column is identical for multiple rows. For example, suppose that the third and fourth leaf nodes of a B-tree structure contain the key value Smith. Suppose further that this value is duplicated six times, as Figure 4-18 illustrates.

The first item on the third leaf page contains the duplicate key value, Smith, and the rowid information for the first physical row in the table that contains the duplicate key value. To conserve space, the second item does not repeat the key value Smith but instead contains just the rowid information. This process continues throughout the page; no other key values are on the leaf, only rowid information.

The first item on the fourth leaf page again contains the duplicated key value and rowid information. Subsequent items contain only rowid information.

Now consider the branch node. The third item in the branch node contains the same key value and rowid as the largest item in the third leaf node and a node pointer to it. The fourth item would contain only a node pointer to the fourth leaf node, thus saving the space of an additional duplicate key value.

Key-Value Locking: To increase concurrency, the database server supports key-value locking in the B-tree index. Key-value locking locks only the value of the key instead of the physical location in the B-tree index.
One of the most important uses for key-value locking is to assure that a unique key remains unique through the end of the transaction that deleted it. Without this protection mechanism, user A might delete a unique key within a transaction, and user B might insert a row with the same key before the transaction commits. This scenario makes rollback by user A impossible. Key-value locking prevents user B from inserting the row until the end of user A’s transaction.

**Adjacent Key Locking:** With Repeatable Read isolation level, the database server is required to protect the read set. The read set consists of the rows that meet the filters in the WHERE clause of the query. To guarantee that the rows do not change, the database server obtains a lock on the index item that is adjacent to the right-most item of the read set.

**Freed Index Pages:** When the database server physically removes an index item from a node and frees an index page, the freed page is reused.

**Filling Indexes:** When you create an index, you can specify how densely or sparsely filled you want the index. The index fill factor is a percentage of each index page that will be filled during the index build. Use the FILLFACTOR option of the CREATE INDEX statement or the FILLFACTOR configuration parameter to set the fill factor. This option is particularly useful for indexes that you do not expect to grow after they are built. For additional information about the FILLFACTOR option of the CREATE INDEX statement, see the IBM Informix Guide to SQL: Syntax.

**Calculating the Length of Index Items:** For data types other than VARCHAR, the length of an index item is calculated by adding the length of the key value plus 5 bytes for each rowid information associated with the key value.

The key values in an index are typically of fixed length. If an index holds the value of one or more columns of the VARCHAR data type, the length of the key value is at least the sum of the length-plus-one of each VARCHAR value in the key.

In Dynamic Server, the maximum length of a key value is 390 bytes. The combined size of VARCHAR columns that make up a key must be less than 390, minus an additional byte for each VARCHAR column. For example, the key length of the index that the database server builds for the following statements equals 390, or 

\[ ((255+1) + (133+1)) \]

```
CREATE TABLE T1 (c1 varchar(255, 10), c2 varchar(133, 10));
CREATE INDEX I1 on T1(c1, c2);
```

**Functional Indexes**

A functional index is one in which all keys derive from the results of a function. If you have a column of pictures, for example, and a function to identify the predominant color, you can create an index on the result of the function. Such an index would enable you to quickly retrieve all pictures having the same predominant color, without re-executing the function.

A functional index uses the same B-tree structure as any other B-tree index. The only difference is that the determining function is applied during an insert or an update whenever the column that is the argument to the function changes. For more information on the nature of functional indexes, refer to your IBM Informix Performance Guide.
To create a functional index, use the CREATE FUNCTION and CREATE INDEX
statements. For more information on these statements, refer to the IBM Informix
Guide to SQL: Syntax.

**Structure of R-Tree Index Pages**

An index structure that relies on one-dimensional ordering of key values does not
work for spatial data; for example, two dimensional geometric shapes such as
circles, squares, and triangles. Efficient retrieval of spatial data, such as the data
used in geographic information systems (GIS) and computer-aided design (CAD)
applications, requires an access method that handles multidimensional data. The
database server implements an R-tree index to access spatial data efficiently. For
information about the structure of index pages, refer to the IBM Informix R-Tree
Index User’s Guide.

**Storage of Simple Large Objects**

This section explains the structures and storage techniques that the database server
uses to store simple large objects (TEXT or BYTE data).

**Structure of a Blobspace**

When you create a blobspace, you can specify the effective size of the data pages,
which are called blobpages. The blobpage size for the blobspace is specified when
the blobspace is created. Blobpage size must be a multiple of page size. (For
information on determining database server page size, see the chapter on
managing disk space in the IBM Informix Administrator’s Guide.) All blobpages
within a blobspace are the same size, but the size of the blobpage can vary
between blobspaces. Blobpage size can be greater than the page size because data
stored in a blobspace is never written to the page-sized buffers in shared memory.

The advantage of customizing the blobpage size is storage efficiency. Within a
blobspace, TEXT and BYTE data is stored in one or more blobpages, but simple
large objects do not share blobpages. Storage is most efficient when the TEXT or
BYTE data is equal to or slightly smaller than the blobpage size.

The blobspace free-map pages and bitmap pages are the size specified as a
database server page, which enables them to be read into shared memory and to
be logged.

When the blobspace is first created, it contains the following structures:

- Blobspace free-map pages
- The blobspace bitmap that tracks the free-map pages
- Unused blobpages

**Structure of a Dbspace Blobpage**

TEXT or BYTE data that is stored in the dbspace is stored in a blobpage. The
structure of a dbspace blobpage is similar to the structure of a dbspace data page.
The only difference is an extra 12 bytes that can be stored along with the TEXT or
BYTE data in the data area.

Simple large objects can share dbspace blobpages if more than one simple large
object can fit on a single page, or if more than one trailing portion of a simple
large object can fit on a single page.
For a discussion of how to estimate the number of dbspace blobpages needed for a specific table, see your IBM Informix Performance Guide.

Each segment of TEXT or BYTE data stored in a dbspace page might be preceded by up to 12 bytes of information that does not appear on any other dbspace page. These extra bytes are overhead.

**Simple-Large-Object Storage and the Descriptor**

Data rows that include TEXT or BYTE data do not include the data in the row itself. Instead, the data row contains a 56-byte descriptor with a forward pointer (rowid) to the location where the first segment of data is stored.

The descriptor can point to one of the following items:
- A page (if the data is stored in a dbspace)
- A blobpage (if the data is stored in a blobspace)
- An optical platter (if you are using the Optical Subsystem)

**Creation of Simple Large Objects**

When a row that contains TEXT or BYTE data is to be inserted, the simple large objects are created first. After the simple large objects are written to disk (or optical medium), the row is updated with the descriptor and inserted.

**Deletion or Insertion of Simple Large Objects**

The database server cannot modify simple large objects. It can only insert or delete them. Deleting a simple large object means that the database server frees the space consumed by the deleted object for reuse.

When TEXT or BYTE data is updated, a new simple large object is created, and the data row is updated with the new blob descriptor. The old image of the row contains the descriptor that points to the obsolete value for the simple large object. The space consumed by the obsolete simple large object is freed for reuse after the update is committed. Simple large objects are automatically deleted if the rows that contain their blob descriptors are deleted. (Blobpages that stored a deleted simple large object are not available for reuse until the logical log that contains the original INSERT record for the deleted simple large object is backed up. For more information, see backing up logical-log files to free blobpages in the chapter on what is the logical log in the IBM Informix Administrator’s Guide.)

**Size Limits for Simple Large Objects**

The largest simple large object that the blob descriptor can accommodate is \( (2^{31} - 1) \), or about 2 gigabytes.

**Blobspace Page Types**

Every blobspace chunk contains three types of pages:
- A blobspace free-map page
- A bitmap page
- Blobpages

**Blobspace Free-Map Page**

The blobspace free-map page identifies unused blobpages so that the database server can allocate them as part of simple-large-object creation. When a blobpage is allocated, the free-map entry for that page is updated. All entries for a single simple large object are linked.
A blobspace free-map page is the size of one database server page. Each entry on a free-map page is 8 bytes, stored as two 32-bit words, as follows:

- The first bit in the first word specifies whether the blobpage is free or used.
- The next 31 bits in the first word identify the logical-log file that was current when this blobpage was written. (This information is needed for logging TEXT or BYTE data.)
- The second word contains the tblspace number associated with the simple large object stored on this page.

The number of entries that can fit on a free-map page depends on the page size of your computer. The number of free-map pages in a blobspace chunk depends on the number of blobpages in the chunk.

**Blobspace Bitmap Page**

The blobspace bitmap page tracks the fullness and number of blobspace free-map pages in the chunk. Each blobspace bitmap page is capable of tracking a quantity of free-map pages that represent more than 4,000,000 blobpages. Each blobspace bitmap page is the size of one page.

**Blobpage**

The blobpage contains the TEXT or BYTE data. Blobpage size is specified by the database server administrator who creates the blobspace. Blobpage size is specified as a multiple of the page size.

**Structure of a Blobspace Blobpage**

The storage strategy used to store simple large objects in a blobspace differs from the dbspace storage strategy. The database server does not combine whole simple large objects or portions of a simple large object on a single blobspace blobpage. For example, if blobspace blobpages are 24 kilobytes each, a simple large object that is 26 kilobytes is stored on two 24-kilobyte pages. The extra 22 kilobytes of space remains unused.

The structure of a blobpage includes a blobpage header, the TEXT or BYTE data, and a page-ending time stamp. The blobpage header includes, among other information, the page-header time stamp and the blob time stamp associated with the forward pointer in the data row. If a simple large object is stored on more than one blobpage, a forward pointer to the next blobpage and another blob time stamp are also included in the blobpage header.

**Sbspace Structure**

An sbspace is similar to a blobspace except that it holds smart large objects.

When an sbspace is created in a database, it contains an sbspace descriptor. Each sbspace chunk contains the following structures:

- Sbspace chunk descriptors
- Chunk free-page list
- An sbspace metadata area (up to one for each chunk)
- Reserved data areas (up to two for each chunk)
- User-data areas (up to two for each chunk)

For best performance, it is recommended that the metadata area be located in the middle of the sbspace. The database server automatically places the metadata area
in the correct location. However, to specify the location of the metadata area, specify the -Mo flag in the onspaces command.

If you do not specify the size of the metadata area in the -Ms flag of the onspaces command, the database server uses the value of AVG_LO_SIZE (defaults to 8 kilobytes) to calculate the size of the metadata area. For more information, see “Creating an Sbspace with the -Df option” on page 14-13.

Normally, you can let the system calculate the metadata size for you. If you want to estimate the size of the metadata area, see the chapter on table performance considerations in the IBM Informix Performance Guide.

Figure 4-19 illustrates the chunk structure of an sbspace as it appears immediately after the sbspace is created. Each reserved area can be allocated to either the user-data or metadata area. Reserved areas are always within the user-data area of the chunk.

A single sbspace chunk

<table>
<thead>
<tr>
<th>Chunk one</th>
<th>Chunk header pages</th>
<th>User data area 1</th>
<th>Reserved area 1</th>
<th>Metadata</th>
<th>User data area 2</th>
<th>Reserved area 2</th>
</tr>
</thead>
</table>

Figure 4-19. A Single Sbspace Chunk

Because the chunk in Figure 4-19 is the first in the sbspace, it contains an sbspace descriptor. The chunk descriptor tblspace in chunk one contains information about chunk one and all chunks added to the sbspace thereafter.

Structure of the Metadata Area

As with the chunk header pages, four areas are exclusive to the first chunk in a sbspace: the sbspace descriptor tblspace, the chunk adjunct tblspace, and the level-1 and level-2 archive tblspaces. The tblspace header section contains a tblspace header for each of these tblspaces (notably excluding the tblspace tblspace). Figure 4-20 shows the layout of the metadata in the single-chunk sbspace.
When you specify the sbspace name in the `oncheck -ps` option, you can display the number of pages allocated and used for each tblspace in the metadata area.

The following describes how the metadata area grows:
- The sbspace descriptor tblspace does not grow.
- The chunk adjunct tblspace grows as chunks are added.
- The LO header tblspace grows as chunks are added.
- The tblspace for user-data free list grows if free spaces in the chunk are heavily fragmented.

**Sbpage Structure**

Each sbpage is composed of three elements: an sbpage header, the actual user data itself, and an sbpage trailer. [Figure 4-21] shows the structure of an sbpage. The sbpage header consists of the standard page header. The sbpage trailer is used to detect an incomplete write on the page and to detect page corruption.

**Multiple Chunk Sbspace**

[Figure 4-22] illustrates a possible configuration for a three-chunk sbspace. In this example, **chunk two** contains no metadata of its own. Metadata information for
chunk two is stored in the metadata area of chunk one.

Time Stamps

The database server uses a time stamp to identify a time when an event occurred relative to other events of the same kind. The time stamp is not a literal time that refers to a specific hour, minute, or second. It is a 4-byte integer that the database server assigns sequentially.

Database and Table Creation: What Happens on Disk

This section explains how the database server stores data related to the creation of a database or table and allocates the disk structures that are necessary to store your data.

Database Creation

After the root dbspace exists, users can create a database. The paragraphs that follow describe the major events that occur on disk when the database server adds a new database.

Disk-Space Allocation for System Catalog Tables

The database server searches the chunk free-list pages in the dbspace, looking for free space in which to create the system catalog tables. For each system catalog table, in turn, the database server allocates eight contiguous pages, the size of the initial extent of each system catalog table. The tables are created individually and do not necessarily reside next to each other in the dbspace. They can be located in different chunks. As adequate space is found for the initial extent of each table, the pages are allocated, and the associated chunk free-list page is updated.
Tracking of System Catalog Tables

The database server tracks newly created databases in the database tblspace, which resides in the root dbspace. An entry describing the database is added to the database tblspace in the root dbspace. (See “Structure of the Database Tblspace” on page 4-7.) For each system catalog table, the database server adds a one-page entry to the tblspace tblspace in the dbspace where the database was built. (See “Structure of the Tblspace Tblspace” on page 4-5.) Figure 4-23 illustrates the relationship between the database tblspace entry and the location of the systables system catalog table for the database.

Figure 4-23. New Databases

For instructions on how to list your databases after you create them, see managing databases in the chapter on managing database-logging status in the IBM Informix Administrator’s Guide.

Table Creation

After the root dbspace exists, and a database has been created, users with the necessary SQL privileges can create a database table. When users create a table, the database server allocates disk space for the table in units called extents (see what is an extent in the chapter on where data is stored in the IBM Informix Administrator’s Guide). The paragraphs that follow describe the major events that occur when the database server creates a table and allocates the initial extent of disk space.

Disk-Space Allocation

The database server searches the chunk free-list pages in the dbspace for contiguous free space equal to the initial extent size for the table. When adequate space is found, the pages are allocated, and the associated chunk free-list page is updated.

If the database server cannot find adequate contiguous space anywhere in the dbspace, it allocates to the table the largest available amount of contiguous space. No error message is returned if an allocation is possible, even when the amount of space allocated is less than the requested amount. If the minimum extent size cannot be allocated, an error is returned. (Extents cannot span two chunks.)

Entry in the Tblspace Tblspace

The database server adds a one-page entry for this table to the tblspace tblspace in this dbspace. The tblspace number assigned to this table is derived from the logical page number in the tblspace tblspace where the table is described. See “Tblspace Numbers” on page 4-6.
The tblspace number indicates the dbspace where the tblspace is located. Tblspace extents can be located in any of the dbspace chunks.

If you must know exactly where the tblspace extents are located, execute the `oncheck -pe` command for a listing of the dbspace layout by chunk.

**Entries in the System Catalog Tables**
The table itself is fully described in entries stored in the system catalog tables for the database. Each table is assigned a table identification number or `tabid`. The `tabid` value of the first user-defined table object in a database is always 100. (The object whose `tabid` = 100 might also be a view, synonym, or a sequence.) For a complete discussion of the system catalog, see the *IBM Informix Guide to SQL: Reference.*

A table can be located in a dbspace that is different than the dbspace that contains the database. The tblspace itself is the sum of allocated extents, not a single, contiguous allocation of space. The database server tracks tblspaces independently of the database.

**Creation of a Temporary Table**
The tasks involved in creating temporary tables are similar to the tasks that the database server performs when it adds a new permanent table. The key difference is that temporary tables do not receive an entry in the system catalog for the database. For more information, see the section defining a temporary table, in the chapter on where data is stored in the *IBM Informix Administrator’s Guide.*
Chapter 5. Interpreting Logical-Log Records

In This Chapter ............................................. 5-1
About Logical-Log Records ................................. 5-1
Transactions That Drop a Table or Index ................. 5-1
Transactions That Are Rolled Back ....................... 5-2
Checkpoints with Active Transactions .................. 5-2
Distributed Transactions .................................. 5-2
Logical-Log Record Structure ............................. 5-3
Logical-Log Record Header ................................. 5-3
Logical-Log Record Types and Additional Columns .... 5-3
Log Record Types for Smart Large Objects ............... 5-14

In This Chapter
To display the logical-log records that the logical-log files contain, use the onlog utility.

This chapter provides the following information:
• Brief guidance on reading logical-log records
• A listing of the different logical-log record types

In general, you do not need to read and interpret your logical-log files. However, onlog output is useful in debugging situations. For example, you might want to use onlog to track a specific transaction or to see what changes the database server made to a specific tblspace. You can also use onlog to investigate the cause of an error that occurs during a rollforward. For more information, see "onlog: Display Logical-Log Contents" on page 10-1

About Logical-Log Records
Most SQL statements generate multiple logical-log records. Interpreting logical-log records is more complicated when the database server records the following events in the logical log:
• A transaction that drops a table or index
• A transaction that rolls back
• A checkpoint in which transactions are still active
• A distributed transaction

The following sections discuss the logical-log records for these events.

Transactions That Drop a Table or Index
Once the database server drops a table or index from a database, it cannot roll back that drop operation. If a transaction contains a DROP TABLE or DROP INDEX statement, the database server handles this transaction as follows:
1. The database server completes all the other parts of the transaction and writes the relevant logical-log records.
2. The database server writes a BEGCOM record to the logical log and the records associated with the DROP TABLE or DROP INDEX (DINDEX, for example).
3. The database server writes a COMMIT record.
If the transaction is terminated unexpectedly after the database server writes the BEGCOM record to the logical log, the database server rolls forward this transaction during recovery because it cannot roll back the drop operation.

Transactions That Are Rolled Back

When a rollback occurs, the database server generates a compensation-log record (CLR) for each record in the logical log that is rolled back. The database server uses the CLRs if a system failure takes place during a rollback. The CLRs provide the database server with information on how far the rollback progressed before the failure occurred. In other words, the database server uses the CLRs to log the rollback.

If a CLR contains the phrase includes next record, the next log record that is printed is included within the CLR log record as the compensating operation. Otherwise, you must assume that the compensating operation is the logical undo of the log record to which the link field of the CLR points.

Checkpoints with Active Transactions

If any transactions are active at the time of a checkpoint, checkpoint records include subentries that describe each of the active transactions using the following columns:

- Log begin (decimal format)
- Transaction ID (decimal format)
- Unique log number (decimal format)
- Log position (hexadecimal format)
- User name

Distributed Transactions

When distributed transactions (transactions that span multiple database servers) generate log records, they are slightly different than nondistributed transactions. You might need to read and interpret them to determine the state of the transaction on both database servers if a failure occurs as a transaction was committing.

The following log records are involved in distributed transactions:

- BEGPREP
- ENDTRANS
- HEURTX
- PREPARE
- TABLOCKS

For more information about this type of logical-log record, see the material on two-phase commit and logical-log records in the IBM Informix Administrator’s Guide.

If you are performing distributed transactions with TP/XA, the database server uses an XAPREPARE record instead of a PREPARE record.
Logical-Log Record Structure

Each logical-log record has header information. Depending on the record type, additional columns of information also appear in the output, as explained in “Logical-Log Record Types and Additional Columns” on page 5-3.

Logical-Log Record Header

Table 5-1 contains sample output to illustrate the header columns that display for a logical-log record.

Table 5-1. Sample Output from onlog

<table>
<thead>
<tr>
<th>addr</th>
<th>len</th>
<th>type</th>
<th>xid</th>
<th>id</th>
<th>link</th>
</tr>
</thead>
<tbody>
<tr>
<td>2c018</td>
<td>32</td>
<td>BEGIN</td>
<td>6</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>2c038</td>
<td>140</td>
<td>HDELETE</td>
<td>6</td>
<td>0</td>
<td>2c018</td>
</tr>
<tr>
<td>2c0c4</td>
<td>64</td>
<td>DELITEM</td>
<td>6</td>
<td>0</td>
<td>2c038</td>
</tr>
<tr>
<td>2c104</td>
<td>40</td>
<td>DELITEM</td>
<td>6</td>
<td>0</td>
<td>2c0c4</td>
</tr>
<tr>
<td>2c12c</td>
<td>72</td>
<td>HDELETE</td>
<td>6</td>
<td>0</td>
<td>2c104</td>
</tr>
<tr>
<td>2c174</td>
<td>44</td>
<td>DELITEM</td>
<td>6</td>
<td>0</td>
<td>2c12c</td>
</tr>
<tr>
<td>2c1a0</td>
<td>72</td>
<td>HDELETE</td>
<td>6</td>
<td>0</td>
<td>2c174</td>
</tr>
<tr>
<td>2c1e8</td>
<td>44</td>
<td>DELITEM</td>
<td>6</td>
<td>0</td>
<td>2c1a0</td>
</tr>
<tr>
<td>2c214</td>
<td>64</td>
<td>HDELETE</td>
<td>6</td>
<td>0</td>
<td>2c1e8</td>
</tr>
<tr>
<td>2c254</td>
<td>56</td>
<td>DELITEM</td>
<td>6</td>
<td>0</td>
<td>2c214</td>
</tr>
<tr>
<td>2c28c</td>
<td>48</td>
<td>DELITEM</td>
<td>6</td>
<td>0</td>
<td>2c254</td>
</tr>
<tr>
<td>2c2bc</td>
<td>24</td>
<td>PERASE</td>
<td>6</td>
<td>0</td>
<td>2c28c</td>
</tr>
<tr>
<td>2c2d4</td>
<td>20</td>
<td>BEGCOM</td>
<td>6</td>
<td>0</td>
<td>2c2bc</td>
</tr>
<tr>
<td>2c2e8</td>
<td>24</td>
<td>ERASE</td>
<td>6</td>
<td>0</td>
<td>2c2d4</td>
</tr>
<tr>
<td>2c300</td>
<td>28</td>
<td>CHFREE</td>
<td>6</td>
<td>0</td>
<td>2c2e8</td>
</tr>
<tr>
<td>2c31c</td>
<td>24</td>
<td>COMMIT</td>
<td>6</td>
<td>0</td>
<td>2c300</td>
</tr>
</tbody>
</table>

Table 5-2 defines the contents of each header column.

Table 5-2. Definition of onlog Header Columns

<table>
<thead>
<tr>
<th>Header Field</th>
<th>Contents</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>addr</td>
<td>Log-record address (log position)</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>len</td>
<td>Record length in bytes</td>
<td>Decimal</td>
</tr>
<tr>
<td>type</td>
<td>Record-type name</td>
<td>ASCII</td>
</tr>
<tr>
<td>xid</td>
<td>Transaction number</td>
<td>Decimal</td>
</tr>
<tr>
<td>id</td>
<td>Logical-log number</td>
<td>Decimal</td>
</tr>
<tr>
<td>link</td>
<td>Link to the previous record in the transaction</td>
<td>Hexadecimal</td>
</tr>
</tbody>
</table>

Logical-Log Record Types and Additional Columns

In addition to the six header columns that display for every record, some record types display additional columns of information. The information that appears varies, depending on record type. Table 5-3 on page 5-4 lists all the record types and their additional columns.
The **Action** column indicates the type of database server action that generated the log entry. The **Additional Columns** and **Format** columns describe what information appears for each record type in addition to the header described in "Logical-Log Record Header" on page 5-3.

**Table 5-3. Logical-Log Record Types**

<table>
<thead>
<tr>
<th>Record Type</th>
<th>Action</th>
<th>Additional Columns</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDCHK</td>
<td>Add chunk.</td>
<td>chunk number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>chunk name</td>
<td>ASCII</td>
</tr>
<tr>
<td>ADDDBS</td>
<td>Add dbspace.</td>
<td>dbspace name</td>
<td>ASCII</td>
</tr>
<tr>
<td>ADDITEM</td>
<td>Add item to index.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key length</td>
<td>Decimal</td>
</tr>
<tr>
<td>ADDLOG</td>
<td>Add log.</td>
<td>log number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>log size (pages)</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pageno</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>ALLOCGENPG</td>
<td>Allocate a generic page.</td>
<td>tblspace ID</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slot flags and length</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page version if delete</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flags, vimage record</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid for previous</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data</td>
<td>ASCII</td>
</tr>
<tr>
<td>ALTERDONE</td>
<td>Alter of fragment complete.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>physical page number</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>previous page</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>logical page number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>version of alter</td>
<td>Decimal</td>
</tr>
<tr>
<td>ALTSPCOLSNEW</td>
<td>Changed columns in an alter table.</td>
<td>number of columns</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>special column list</td>
<td>array</td>
</tr>
<tr>
<td>ALTSPCOLSOLD</td>
<td>Changed columns in an alter table.</td>
<td>number of columns</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>special column list</td>
<td>array</td>
</tr>
<tr>
<td>BADIDX</td>
<td>Bad index</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>BEGCOM</td>
<td>Begin commit.</td>
<td>(None)</td>
<td>(None)</td>
</tr>
<tr>
<td>BEGIN</td>
<td>Begin work.</td>
<td>date</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>time</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SID</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>user</td>
<td>ASCII</td>
</tr>
<tr>
<td>BEGPREP</td>
<td>Written by the coordinator database server to record the start of the two-phase commit protocol.</td>
<td>flags</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Value is 0 in a distributed transaction.)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of participants</td>
<td>Decimal</td>
</tr>
<tr>
<td>Record Type</td>
<td>Action</td>
<td>Additional Columns</td>
<td>Format</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>---------------------</td>
<td>--------</td>
</tr>
<tr>
<td>BEGWORK</td>
<td>Begin a transaction.</td>
<td>begin transaction time</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>user ID</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>process ID</td>
<td>Decimal</td>
</tr>
<tr>
<td>BFRMAP</td>
<td>Simple-large-object free-map change.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bpageno</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>status</td>
<td>USED/FREE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>log ID</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>prev page</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>BLDCL</td>
<td>Build tblspace.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fextsize</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nextsize</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>row size</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ncolumns</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>table name</td>
<td>ASCII</td>
</tr>
<tr>
<td>BMAPFULL</td>
<td>Bitmap modified to prepare for alter.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bitmap page num</td>
<td>Decimal</td>
</tr>
<tr>
<td>BMAP2TO4</td>
<td>2-bit bitmap altered to two 4-bit bitmaps.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2-bit bitmap page number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flags</td>
<td>Decimal</td>
</tr>
<tr>
<td>BSPADD</td>
<td>Add blobspace.</td>
<td>blobspace name</td>
<td>ASCII</td>
</tr>
<tr>
<td>BTCPYBCK</td>
<td>Copy back child key to parent.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>parent logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>child logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slot</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowoff</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key number</td>
<td>Decimal</td>
</tr>
<tr>
<td>BTMERGE</td>
<td>Merge B-tree nodes.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>parent logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>right logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left slot</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left rowoff</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>right slot</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>right rowoff</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key number</td>
<td>Decimal</td>
</tr>
<tr>
<td>Record Type</td>
<td>Action</td>
<td>Additional Columns</td>
<td>Format</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>---------------------</td>
<td>--------</td>
</tr>
<tr>
<td>BTSHUFFL</td>
<td>Shuffle B-tree nodes.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>parent logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>right logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left slot</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left rowoff</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flags</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>BTSPLIT</td>
<td>Split B-tree node.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>parent logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>right logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>infinity logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rootleft logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>midsplit</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key length</td>
<td>Decimal</td>
</tr>
<tr>
<td>CDINDEX</td>
<td>Create detached index.</td>
<td>database name</td>
<td>ASCII</td>
</tr>
<tr>
<td></td>
<td></td>
<td>owner</td>
<td>ASCII</td>
</tr>
<tr>
<td></td>
<td></td>
<td>table name</td>
<td>ASCII</td>
</tr>
<tr>
<td></td>
<td></td>
<td>index name</td>
<td>ASCII</td>
</tr>
<tr>
<td>CDR</td>
<td>Captures the set of table columns modified by an update statement such as a bitvector. This log record allows Enterprise Replication to capture only the changed data to avoid transmitting the unchanged columns to a target site.</td>
<td>name of CDR record</td>
<td>ASCII</td>
</tr>
<tr>
<td></td>
<td></td>
<td>partition number</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bitvector</td>
<td>Binary</td>
</tr>
</tbody>
</table>

Sample **onlog** output for CDR log record:
```
adr len type xid id link name partno bitvector
40 36 CDR 14 0 18 UPDCOLS 10009a 0000001101001100
```

<table>
<thead>
<tr>
<th>CHALLOC</th>
<th>Chunk extent allocation.</th>
<th>pageno</th>
<th>Hexadecimal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>size</td>
<td>Hexadecimal</td>
</tr>
</tbody>
</table>
### Table 5-3. Logical-Log Record Types (continued)

<table>
<thead>
<tr>
<th>Record Type</th>
<th>Action</th>
<th>Additional Columns</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHCOMBINE</td>
<td>Chunk extent combine.</td>
<td>pageno</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>CHFREE</td>
<td>Chunk extent free.</td>
<td>pageno</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>size</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>CHKADJUP</td>
<td>Update chunk adjunct on disk. The database server writes this record</td>
<td>chunk number</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td>when it moves space from the reserved area to the metadata or</td>
<td>ud1_start_page</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td>user-data area or when the user adds an sbspace chunk.</td>
<td>ud1_size</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>md_start_page</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>md_size</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ud2_start_page</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ud2_size</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flags</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>CHPHYLOG</td>
<td>Change physical-log location.</td>
<td>pageno</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>size in kilobytes</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dbspace name</td>
<td>ASCII</td>
</tr>
<tr>
<td>CHRESERV</td>
<td>Reserve extent for metadata stealing. This record is written when you</td>
<td>chunk number</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td>add an sbspace chunk.</td>
<td>page number</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>length</td>
<td>Integer</td>
</tr>
<tr>
<td>CHSPLIT</td>
<td>Chunk extent split.</td>
<td>pageno</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>CINDEX</td>
<td>Create index.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>low rowid</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>high rowid</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>index descriptor</td>
<td>ASCII</td>
</tr>
<tr>
<td>COARSELOCK</td>
<td>Coarse-grain locking</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>old coarse-locking flag value</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new coarse-locking flag value</td>
<td>Decimal</td>
</tr>
<tr>
<td>CKPOINT</td>
<td>Checkpoint.</td>
<td>max users</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of active transactions</td>
<td>Decimal</td>
</tr>
<tr>
<td>CLR</td>
<td>Compensation-log record; created during a rollback.</td>
<td>(None)</td>
<td>(None)</td>
</tr>
<tr>
<td>CLUSIDX</td>
<td>Create clustered index.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key number</td>
<td>Decimal</td>
</tr>
<tr>
<td>COLREPAI</td>
<td>Adjust BYTE, TEXT, or VARCHAR column.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of columns adjusted</td>
<td>Decimal</td>
</tr>
<tr>
<td>COMMIT</td>
<td>Commit work.</td>
<td>date</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>time</td>
<td>Decimal</td>
</tr>
<tr>
<td>Record Type</td>
<td>Action</td>
<td>Additional Columns</td>
<td>Format</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>---------------------</td>
<td>--------</td>
</tr>
<tr>
<td>COMTAB</td>
<td>Compact slot table on a page.</td>
<td>logical page number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number slots moved</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>compressed slot pairs</td>
<td>ASCII</td>
</tr>
<tr>
<td>COMWORK</td>
<td>End a transaction and commit work.</td>
<td>end transaction time</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>begin transaction time</td>
<td>Decimal</td>
</tr>
<tr>
<td>DELETE</td>
<td>Delete before-image.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>DELITEM</td>
<td>Delete item from index.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key length</td>
<td>Decimal</td>
</tr>
<tr>
<td>DERASE</td>
<td>Drop tblspace in down dbspace.</td>
<td>tblspace number</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>table lock number</td>
<td>Decimal</td>
</tr>
<tr>
<td>DINDEX</td>
<td>Drop index.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>key number</td>
<td>Decimal</td>
</tr>
<tr>
<td>DRPBSP</td>
<td>Drop blobspace.</td>
<td>blobspace name</td>
<td>ASCII</td>
</tr>
<tr>
<td>DRPCHK</td>
<td>Drop chunk.</td>
<td>chunk number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>chunk name</td>
<td>ASCII</td>
</tr>
<tr>
<td>DRPDBS</td>
<td>Drop dbspace.</td>
<td>dbspace name</td>
<td>ASCII</td>
</tr>
<tr>
<td>DRPLOG</td>
<td>Drop log.</td>
<td>log number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>log size (pages)</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pageno</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>ENDTTRANS</td>
<td>Written by both the coordinator and participant database servers to record the end of the transaction. ENDTTRANS instructs the database server to remove the transaction entry from its shared-memory transaction table and close the transaction.</td>
<td>(None)</td>
<td>(None)</td>
</tr>
<tr>
<td>ERASE</td>
<td>Drop tblspace.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
</tbody>
</table>
Table 5-3. Logical-Log Record Types (continued)

<table>
<thead>
<tr>
<th>Record Type</th>
<th>Action</th>
<th>Additional Columns</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>FREE_RE</td>
<td>Allocate extent from reserve extent to metadata or user-data area of an sbspace chunk.</td>
<td>chunk number</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page number</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>length</td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flag</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>HDELETE</td>
<td>Delete home row.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td>HEURTX</td>
<td>Written by a participant database server to record a heuristic decision to roll back the transaction. It should be associated with a standard ROLLBACK record indicating that the transaction was rolled back.</td>
<td>flag</td>
<td>Hexadecimal (Value is always 1.)</td>
</tr>
<tr>
<td>HINSERT</td>
<td>Home row insert.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td>HUPAFT</td>
<td>Home row update, after-image.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td>HUPBEF</td>
<td>Home row update, before-image.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td>In addition, the flag field of the HUPBEF header may include the following values:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>LM_PREVLSN</strong></td>
<td>Confirms that an LSN exists.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>LM_FIRSTUPD</strong></td>
<td>Confirms that this is the first update for this rowID by this transaction.</td>
<td></td>
</tr>
<tr>
<td>HUPDATE</td>
<td>If the home row update before-images and after-images can both fit into a single page, the database server writes a single HUPDATE record.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td>In addition, the flag field of the HUPDATE log may include the following values:</td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td><strong>LM_PREVLSN</strong></td>
<td>Confirms that an LSN exists.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>LM_FIRSTUPD</strong></td>
<td>Confirms that this is the first update for this rowID by this transaction.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>forward ptr rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>old slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of pieces</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LSN (optional)</td>
<td>Decimal</td>
</tr>
<tr>
<td>Record Type</td>
<td>Action</td>
<td>Additional Columns</td>
<td>Format</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>IDXFLAGS</td>
<td>Index flags.</td>
<td>tblspace ID key number</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>INSERT</td>
<td>Insert after-image.</td>
<td>tblspace ID rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>ISOSPCOMMIT</td>
<td>Log an isolated save-point commit.</td>
<td>end transaction time begin transaction time</td>
<td>Decimal</td>
</tr>
<tr>
<td>LCKLVL</td>
<td>Locking mode (page or row).</td>
<td>tblspace ID old lockmode new lockmode</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>LG_ADDBPOOL</td>
<td>Add a buffer pool online.</td>
<td>page size in bytes number of buffers in the pool number of lru queues percent of lru_max_dirty percent of lru_min_dirty</td>
<td>Decimal</td>
</tr>
<tr>
<td>PTRUNCATE</td>
<td>Identifies an intention to truncate a table. The partitions are marked to be dropped or reused, according to the specified command option.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>TRUNCATE</td>
<td>TRUNCATE has freed the extents and the transaction will be committed.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>MVIDXND</td>
<td>Index node moved to allow for 2-bit to 4-bit bitmap conversion.</td>
<td>tblspace ID old page number new page number parent page number parent slot number parent slot offset key number</td>
<td>Decimal</td>
</tr>
<tr>
<td>PBDELETE</td>
<td>Delete tblspace blobpage.</td>
<td>bpageno status unique ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>PBINSERT</td>
<td>Insert tblspace blobpage.</td>
<td>bpageno tblspace ID rowid slotlen pbrowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>PDINDEX</td>
<td>Predrop index.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>PGALTER</td>
<td>Page altered in place.</td>
<td>tblspace ID physical page number</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>Record Type</td>
<td>Action</td>
<td>Additional Columns</td>
<td>Format</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------------------</td>
<td>----------</td>
</tr>
<tr>
<td>PGMODE</td>
<td>Page mode modified in bitmap.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>logical page number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>old mode</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new mode</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>PERASE</td>
<td>Preerase old file. Mark a table that is to be dropped. The database server frees the space on the commit.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>PNGPALIGN8</td>
<td>Use the pages in this tblspace as generic pages.</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>PNLOCKID</td>
<td>Change tblspaces lockid.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>old lock ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new lock ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>PNSIZES</td>
<td>Set tblspace extent sizes.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fextsize</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nextsize</td>
<td>Decimal</td>
</tr>
<tr>
<td>PREPARE</td>
<td>Written by a participant database server to record the ability of the participant to commit the transaction, if so instructed.</td>
<td>DBSERVERNAME of coordinator</td>
<td>ASCII</td>
</tr>
<tr>
<td>PTADESC</td>
<td>Add alter description information.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>physical page number</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>logical page number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of columns added</td>
<td>Decimal</td>
</tr>
<tr>
<td>PTALTER</td>
<td>Alter of fragment begun.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>physical page number</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>previous page</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>logical page number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>alter desc page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>num columns added</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>version of alter</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>added rowsize</td>
<td>Decimal</td>
</tr>
<tr>
<td>PTALTNEWKEYD</td>
<td>Update key descriptors in a tblspace header after an alter table command.</td>
<td>bytes in key descriptor</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data in key descriptor</td>
<td>ASCII</td>
</tr>
<tr>
<td>PTALTOLDKEYD</td>
<td>Update key descriptors after an alter table command.</td>
<td>bytes in key descriptor</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data in key descriptor</td>
<td>ASCII</td>
</tr>
<tr>
<td>PTCOLUMN</td>
<td>Add special columns to fragment.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of columns</td>
<td>Decimal</td>
</tr>
<tr>
<td>PTEXTEND</td>
<td>Tblspace extend.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>last logical page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>first physical page</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>Record Type</td>
<td>Action</td>
<td>Additional Columns</td>
<td>Format</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>-------------------</td>
<td>--------</td>
</tr>
<tr>
<td>PTRENAME</td>
<td>Rename table.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>old table name</td>
<td>ASCII</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new table name</td>
<td>ASCII</td>
</tr>
<tr>
<td>RDELETE</td>
<td>Remainder page delete.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hrowid (optional)</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>poffset (optional)</td>
<td>Decimal</td>
</tr>
<tr>
<td>RENDBS</td>
<td>Rename dbspace.</td>
<td>new dbspace name</td>
<td>ASCII</td>
</tr>
<tr>
<td>REVERT</td>
<td>Logs the reversion of a database space to a database space of an earlier version.</td>
<td>type of reversion event</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>arg1</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>arg2</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>arg3</td>
<td>Decimal</td>
</tr>
<tr>
<td>RINSERT</td>
<td>Remainder page insert.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hrowid (optional)</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>poffset (optional)</td>
<td>Decimal</td>
</tr>
<tr>
<td>ROLBACK</td>
<td>Rollback work.</td>
<td>date</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>time</td>
<td>Decimal</td>
</tr>
<tr>
<td>ROLWORK</td>
<td>End a transaction and roll back work.</td>
<td>end transaction time</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>begin transaction time</td>
<td>Decimal</td>
</tr>
<tr>
<td>RSVEXTEND</td>
<td>Logs the extension to the reserved pages.</td>
<td>number of pages</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>physical page number of extent</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>RTREE</td>
<td>Logs inserts and deletions for R-tree index pages. (Other operations on R-tree indexes are physically logged.) The record subtypes are:</td>
<td>record subtype</td>
<td>ASCII</td>
</tr>
<tr>
<td></td>
<td></td>
<td>index page rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tuple length</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>base table rowid</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>base table fragid</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>delete flag</td>
<td>Decimal</td>
</tr>
<tr>
<td>RUPAFT</td>
<td>Remainder page update, after-image.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td>RUPBEF</td>
<td>Remainder page update, before-image.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hrowid (optional)</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>poffset (optional)</td>
<td>Decimal</td>
</tr>
<tr>
<td>Record Type</td>
<td>Action</td>
<td>Additional Columns</td>
<td>Format</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>---------------------</td>
<td>--------</td>
</tr>
<tr>
<td><strong>RUPDATE</strong></td>
<td>If the remainder page update before-images and after-images can both fit into a single page, the database server writes a single RUPDATE record.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>forward ptr rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>old slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new slotlen</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of pieces</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hrowid (optional)</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>poffset (optional)</td>
<td>Decimal</td>
</tr>
<tr>
<td><strong>SBLOB</strong></td>
<td>Indicates a subsystem log record for a smart large object.</td>
<td>Varies</td>
<td>Varies</td>
</tr>
<tr>
<td></td>
<td>The various record subtypes are:</td>
<td>For more information, see &quot;Log Record Types for Smart Large Objects&quot; on page 5-12</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CHALLOC</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CHCOMBINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CHFREE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CHSPLIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CREATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DELETES</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>EXTEND</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>HDRUDP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PDELETE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PTRUNC</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>REFCOUNT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDINSERT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDINSERT_LT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDUPAFT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDUPAFT_LT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDUPAFT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDUPAFT_LT</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDWRITE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UDWRITE_LT</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>SYNC</strong></td>
<td>Written to a logical-log file if that log file is empty and administrator instructs the database server to switch to next log file.</td>
<td>(None)</td>
<td>(None)</td>
</tr>
<tr>
<td><strong>TABLOCKS</strong></td>
<td>Written by either a coordinator or a participant database server. It is associated with either a BEGPREP or a PREPARE record and contains a list of the locked tblspaces (by tblspace number) held by the transaction. (In a distributed transaction, transactions are shown as the owners of locks.)</td>
<td>number of locks</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tblspace number</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td><strong>UDINSERT</strong></td>
<td>Append new user data.</td>
<td>chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page within chunk</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>offset within page</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data length</td>
<td>Hexadecimal</td>
</tr>
</tbody>
</table>
Table 5-3. Logical-Log Record Types (continued)

<table>
<thead>
<tr>
<th>Record Type</th>
<th>Action</th>
<th>Additional Columns</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDUPAFT</td>
<td>Update user data after-image if a UDWRITE is too expensive.</td>
<td>chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page within chunk</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>offset within page</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data length</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>UDUPBEF</td>
<td>Update user-data before-image if a UDWRITE is too expensive.</td>
<td>chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page within chunk</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>offset within page</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data length</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>UDWRITE</td>
<td>Update user data (difference image).</td>
<td>chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page within chunk</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>offset within chunk</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>length before write</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>length after write</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>UNDO</td>
<td>Header record to a series of transactions to be rolled back.</td>
<td>count</td>
<td>Decimal</td>
</tr>
<tr>
<td>UNDOBLDC</td>
<td>This record is written if a CREATE TABLE statement should be rolled back but cannot be because the relevant chunk is down. When the log file is replayed, the table will be dropped.</td>
<td>tblspace number</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>UNIQID</td>
<td>Logged when a new serial value is assigned to a row.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>unique ID</td>
<td>Decimal</td>
</tr>
<tr>
<td>UPDAFT</td>
<td>Update after-image.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>UPDBEF</td>
<td>Update before-image.</td>
<td>tblspace ID</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rowid</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>XAPREPAIR</td>
<td>Participant can commit this XA transaction.</td>
<td>(None)</td>
<td>(None)</td>
</tr>
</tbody>
</table>

Log Record Types for Smart Large Objects

All smart-large-object log records are the SBLOB type. Each smart-large-object log record contains six header columns, described in "Logical-Log Record Header" on page 5-3, the record subtype; and additional information. The information that appears varies, depending on record subtype.

Table 5-4 lists all the smart-large-object record types. The Subtype column describes the smart-large-object record type. The Action column indicates the type of database server action that generated the log entry. The Additional Columns and Format columns describe what information appears for each record type.
<table>
<thead>
<tr>
<th>Record Subtype</th>
<th>Action</th>
<th>Additional Columns</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHALLOC</td>
<td>Allocate chunk extent.</td>
<td>extent [chk, page, len]</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>flags</td>
<td>Hexadecimal</td>
</tr>
<tr>
<td>CHCOMBINE</td>
<td>Combine two pages in the user-data extent list.</td>
<td>chunk number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>first page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>second page</td>
<td>Decimal</td>
</tr>
<tr>
<td>CHFREE</td>
<td>Frees chunk extent.</td>
<td>extent [chk, page, len]</td>
<td>Decimal</td>
</tr>
<tr>
<td>CHSPLIT</td>
<td>Split a page in the user-data extent list.</td>
<td>chunk number</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>UDFET page to split</td>
<td>Decimal</td>
</tr>
<tr>
<td>CREATE</td>
<td>Create smart large object.</td>
<td>smart-large-object ID [sbs, chk, page, oid]</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of extents in lomaphdr</td>
<td>Decimal</td>
</tr>
<tr>
<td>DELETE</td>
<td>Delete a smart large object at commit.</td>
<td>smart-large-object ID [sbs, chk, page, oid]</td>
<td>Decimal</td>
</tr>
<tr>
<td>EXTEND</td>
<td>Add extent to an extent list of a smart large object.</td>
<td>smart-large-object ID [sbs, chk, page, oid]</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>extent [chk, page, len]</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>lomap overflow page number</td>
<td>Decimal</td>
</tr>
<tr>
<td>HDRUPD</td>
<td>Update smart-large-object header page.</td>
<td>smart-large-object ID [sbs, chk, page, oid]</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>old EOF offset</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new EOF offset</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>old times</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new times</td>
<td>Decimal</td>
</tr>
<tr>
<td>PDELETE</td>
<td>Queue a smart large object for deletion at commit.</td>
<td>smart-large-object ID [sbs, chk, page, oid]</td>
<td>Decimal</td>
</tr>
<tr>
<td>PTRUNC</td>
<td>Queue a smart large object for truncation at commit.</td>
<td>smart-large-object ID [sbs, chk, page, oid]</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>old offset</td>
<td>String</td>
</tr>
<tr>
<td></td>
<td></td>
<td>new offset</td>
<td>String</td>
</tr>
<tr>
<td>REFCOUNT</td>
<td>Increment or decrement the reference count of a smart large object.</td>
<td>smart-large-object ID [sbs, chk, page, oid]</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 if increment; 0 if decrement</td>
<td>Decimal</td>
</tr>
<tr>
<td>UDINSERT, UDINSERT_LT</td>
<td>Append new user data.</td>
<td>chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page within chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>offset within page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data length</td>
<td>Decimal</td>
</tr>
<tr>
<td>UDUPAFT, UDUPAFT_LT</td>
<td>Update user-data after-image if a UDWRITE is too expensive.</td>
<td>chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>page within chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>offset within page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data length</td>
<td>Decimal</td>
</tr>
</tbody>
</table>
### Table 5-4. Record Subtypes for Smart Large Objects (continued)

<table>
<thead>
<tr>
<th>Record Subtype</th>
<th>Action</th>
<th>Additional Columns</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDUPBEF,</td>
<td>Update user-data beforeimage if a UDWRITE is too expensive.</td>
<td>chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td>UDUPBEF_LT</td>
<td></td>
<td>page within chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>offset within page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>data length</td>
<td>Decimal</td>
</tr>
<tr>
<td>UDWRITE,</td>
<td>Update user data (difference image).</td>
<td>chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td>UDWRITE_LT</td>
<td></td>
<td>page within chunk</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>offset within page</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>length before write</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>length after write</td>
<td>Decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>number of different image pieces</td>
<td>Decimal</td>
</tr>
</tbody>
</table>

For an example of smart-large-object records in onlog output, see smart-large-object log records in the chapter on what is the logical log in the IBM Informix Administrator’s Guide.

Figure 5-1 shows an example of smart-large-object records in onlog output. The first two records show that an extent was freed. The next group of records, flanked by BEGIN and COMMIT, shows the allocation of storage and creation of the smart large objects.

```
addr len type xid id link subtype specific-info
4e8428 40 SBLOB 8 0 4e7400 CHFREE (2,53,421)
4e8450 40 SBLOB 8 0 4e8428 CHFREE (2,579,421)
c8018 40 BEGIN 8 3 0 07/13/98 10:23:04 34 informix
4e8428 40 SBLOB 8 0 4e7400 CHFREE (2,53,421)
c8040 264 SBLOB 8 0 c8018 CREATE [2,2,1,900350517] 10
4e8450 40 SBLOB 8 0 4e8428 CHFREE (2,579,421)
c8148 44 SBLOB 8 0 c8040 CHALLOC (2,53,8) 0x1
c8174 68 SBLOB 8 0 c8148 EXTEND [2,2,1,900350517] (2,53,8) -1
c81b8 264 SBLOB 8 0 c8174 CREATE [2,2,2,900350516] 10
4e8428 40 SBLOB 8 0 4e7400 CHFREE (2,53,421)
c82c0 44 SBLOB 8 0 c81b8 CHALLOC (2,61,1) 0x1
c82ec 68 SBLOB 8 0 c82c0 EXTEND [2,2,2,900350518] (2,61,1) -1
c8330 56 SBLOB 8 0 c82ec REFCOUNT [2,2,1,900350517] 1
4e8428 40 SBLOB 8 0 4e7400 CHFREE (2,53,421)
c8368 56 SBLOB 8 0 c8330 REFCOUNT [2,2,2,900350518] 1
4e8450 40 SBLOB 8 0 4e8428 CHFREE (2,579,421)
c83a0 36 COMMIT 8 0 c8368 07/13/98 10:23:05
4e8428 40 SBLOB 8 0 4e7400 CHFREE (2,53,421)
c83ec 264 SBLOB 8 0 c83a0 CREATE [2,2,3,900350519] 10
4e8450 40 SBLOB 8 0 4e8428 CHFREE (2,579,421)
c84f4 44 SBLOB 8 0 c83ec CHALLOC (2,62,1) 0x1
c8520 68 SBLOB 8 0 c84f4 EXTEND [2,2,3,900350519] (2,62,1) -1
c8564 56 SBLOB 8 0 c8520 REFCOUNT [2,2,3,900350519] 1
4e8428 40 SBLOB 8 0 4e7400 CHFREE (2,53,421)
c859c 36 COMMIT 8 0 c8564 07/13/98 10:23:05
```

Figure 5-1. Smart-Large-Object Records in onlog Output
Part 2. Administrative Utilities
Chapter 6. Overview of Utilities

In This Chapter

- Complete List of Utilities
- Obtaining Utility Version Information
- Syntax of Utility-Specific Options
- Multibyte Characters (GLS)
- OpenAdmin Tool for IDS
- IBM Informix Server Administrator
- Server Studio

This chapter provides reference material for the Informix database server utilities. These utilities allow you to perform administrative tasks directly from the command line. For a complete listing of utilities, see your IBM Informix Getting Started Guide.

You can use the following utilities:

- IBM Informix Server Administrator (ISA)
- ON–Bar
- oncheck
- ondblog
- oninit
- onlog
- onmode
- ON–Monitor
- onparams
- onspaces
- onstat
- ontape
- OpenAdmin Tool for IDS

The database server must be online before you execute a utility, with the following exceptions:

- oninit
- Some onlog options
- Some oncheck options

Note: When using utilities, do not use the UNIX command CTRL-C to send an interrupt signal to a process because it might produce an error.

Complete List of Utilities

The appendix in your IBM Informix Getting Started Guide contains a quick reference to all utilities and their options.
Obtaining Utility Version Information

Many Informix command-line utilities allow you to obtain version information using -V and -version options. You use the -V and -version options primarily for debugging. When a Technical Support representative asks for the version number, you can use the -V and -version options to find the information.

The -V option displays the software version number and the serial number.

The -version option extends the -V option to display additional information on the build operation system, build number, and build date.

Syntax of Utility-Specific Options
The following syntax diagram illustrates the -V and -version options

```
utility  utility specific options
        -V
        -version
```

The -V and -version options cannot be used with any other utility options. For example, the onstat -version command might display the following output.

```
onstat -version
Program: onstat
Build Version: 11.10.FC1
Build Host: connla
Build OS: SunOS 5.6
Build Number: 009
Build Date: Sat Aug 11 03:38:27 CDT 2007
GLS Version: glslib-4.50.xC2
```

The onstat -V command might display the following information:

```
IBM Informix Dynamic Server Version 11.10.FC1     Software Serial Number
RDS#N000000
```

Multibyte Characters (GLS)

The database server utilities support multibyte command-line arguments. For a complete list of the utilities that support multibyte command-line arguments, see the IBM Informix GLS User’s Guide.

OpenAdmin Tool for IDS

OpenAdmin Tool for IDS is a PHP-based Web browser administration tool that can administer multiple database server instances using a single installation on a Web server. You access the Web server through any browser to administer all your database servers. You can perform the following tasks with OpenAdmin:

- Add new connections to servers and server groups
- View server information on a map
- Customize the help system to add your own content to help topics
- Configure the display to change the sort order of reports by clicking on column headings
- Set up RSS feeds to notify you of specified events
- Manage dbspaces, chunks, and recovery logs
• Monitor performance statistics, including recent SQL statements, combine graphs and reports to create custom reports
• View the online message log and the ON-Bar activity log
• Execute ad hoc or saved SQL statements
• View database, table, and column information
• Monitor high-availability clusters: HDR servers, Shared Disk secondary servers, and Remote Standalone secondary servers
• View information about executed SQL Administration API commands

OpenAdmin is an open-source program that you can download from this Web site: [http://www.ibm.com/software/data/informix/downloads.html](http://www.ibm.com/software/data/informix/downloads.html)

### IBM Informix Server Administrator

IBM Informix Server Administrator (ISA) allows a DBA to manage Informix database servers by executing Informix commands from any web browser. You do not need to be familiar with the syntax and format of database server commands. ISA presents the command output in an easy-to-read format.


With ISA, you can perform these database server administrative tasks:
• Change configuration parameters temporarily or permanently.
• Use Server Setup to configure or reconfigure the database server.
• Change the database server mode.
• Modify connectivity information in the sqlhosts file.
• Check dbspaces, blobspaces, and sbspaces.
• Manage logical logs and physical logs.
• Examine and modify memory usage.
• Read the message log.
• Back up and restore dbspaces, blobspaces, and sbspaces.
• Run various onstat commands to monitor performance.
• Enter SQL statements and examine database schemas.
• Add and remove chunks, dbspaces, blobspaces, sbspaces.
• Examine and manage user sessions.
• Examine and manage virtual processors (VPs).
• Use the High-Performance Loader (HPL), dbimport, and dbexport.
• Manage Enterprise Replication.
• Manage a MaxConnect server.
• Set up primary and secondary database servers for High-Availability Data Replication.
• Use the following utilities: dbaccess, dbschema, onbar, oncheck, ondblog, oninit, onlog, onmode, onparams, onspaces, onstat, onpladm.
• Enter any Informix utility, UNIX shell command, or Windows command.
Server Studio

Server Studio by AGS is a CD included in the IBM Informix Dynamic Server bundle. It provides a collection of tools for DBAs and developers for performing common database tasks. Server Studio contains the following modules:

- Object Explorer
- Schema Editor
- SQL Editor

Server Studio offers several additional modules that are available on a 30-day trial basis. For more information, visit the AGS Web site at [http://www.agsltd.com](http://www.agsltd.com).
Chapter 7. The oncheck Utility

In This Chapter

In This Chapter shows you how to use the following oncheck options and functionalities:

- oncheck -cc and -pc: Check system catalog tables on page 7-8
- oncheck -cd and -cD: Check pages on page 7-9
- oncheck -ce, -pE: Check the chunk-free list on page 7-10
- oncheck -ci and -cI: Check index node links on page 7-10
- oncheck -cr and -cR: Check reserved pages on page 7-12
- oncheck -cs, -cs, -ps, -ps: Check and display sbspaces on page 7-12
- oncheck -pB: Display blobspace statistics on page 7-13
- oncheck -pd and -pD: Display rows in hexadecimal format on page 7-13
- oncheck -pk, -pK, -pL, -pL: Display index information on page 7-15
- oncheck -pp and -pP: Display the contents of a logical page on page 7-16
- oncheck -pr and -P: Display reserved-page information on page 7-18
- oncheck -pt and -P: Display blobspages for a table or fragment on page 7-18
- Turn On Locking with -x on page 7-20
- Send Special Arguments to the Access Method with -u on page 7-20
- Return Codes on Exit on page 7-20

oncheck Check-and-Repair

The oncheck utility can repair the following types of disk structures:

- Partition page statistics
- Bitmap pages
- Partition blobpages
- Blobspace blobpages
Indexes
Sbspaces
Metadata partitions for sbspaces

If oncheck detects inconsistencies in other structures, messages alert you to these inconsistencies, but oncheck cannot resolve the problem. For more information, see the chapter on consistency checking in the IBM Informix Administrator’s Guide and Chapter 4, “Disk Structures and Storage,” on page 4-1.

What Does Each Option Do?
As Table 7-1 on page 7-2 shows, the oncheck options fall into three categories: check, repair, and display. The display or print options (those prefixed with the letter p) are identical in function to the -c options, except that the -p options display additional information about the data that is being checked as the oncheck utility executes. You cannot combine oncheck option flags except as the following paragraphs describe.

In general, the -c options check for consistency and display a message on the screen only if they find an error or inconsistency.

Any user can execute the check options. On UNIX platforms, you must be user informix or root to display database data or initiate . On Windows, you must be a member of the Informix-Admin group to display database data or initiate repair options.

Table 7-1 associates oncheck options with their function. It also shows the Administrative API command strings that are equivalent to the oncheck -c options.

Table 7-1. oncheck Options and Their Function

<table>
<thead>
<tr>
<th>Object</th>
<th>Check</th>
<th>Administrative API command string</th>
<th>Repair</th>
<th>Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blobspace simple large objects</td>
<td>-cc</td>
<td></td>
<td></td>
<td>-pB</td>
</tr>
<tr>
<td>System catalog tables</td>
<td>-cc</td>
<td></td>
<td></td>
<td>-pc</td>
</tr>
<tr>
<td>Data rows, no simple large objects or smart large objects</td>
<td>-cd</td>
<td></td>
<td></td>
<td>-pd</td>
</tr>
<tr>
<td>Data rows, simple large objects but no smart large objects</td>
<td>-cD</td>
<td></td>
<td></td>
<td>-pD</td>
</tr>
<tr>
<td>Table with a user-defined access method</td>
<td>-cd, -cD</td>
<td>CHECK DATA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chunks and extents</td>
<td>-ce</td>
<td>CHECK EXTENTS</td>
<td></td>
<td>-pe</td>
</tr>
<tr>
<td>Index (key values)</td>
<td>-ci, -cix</td>
<td>-ci -y, -pk -y, -pkx -y</td>
<td></td>
<td>-pk</td>
</tr>
<tr>
<td>Index (keys plus rowids)</td>
<td>-cil, -cix</td>
<td>-cl -y, -PK -y, -pKx -y</td>
<td></td>
<td>-pK</td>
</tr>
<tr>
<td>Index with a user-defined access method</td>
<td>-ci, -ci</td>
<td>-pl -y, -plx -y</td>
<td></td>
<td>-pl</td>
</tr>
<tr>
<td>Index (leaf key values)</td>
<td>-ci, -cI</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 7-1. oncheck Options and Their Function (continued)

<table>
<thead>
<tr>
<th>Object</th>
<th>Check</th>
<th>Administrative API command string</th>
<th>Repair</th>
<th>Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index (leaf keys plus rowids)</td>
<td></td>
<td>-pL, -y</td>
<td>-pL</td>
<td></td>
</tr>
<tr>
<td>Pages (by table or fragment)</td>
<td></td>
<td>-pL, -y</td>
<td>-PP</td>
<td></td>
</tr>
<tr>
<td>Pages (by chunk)</td>
<td></td>
<td></td>
<td>-pP</td>
<td></td>
</tr>
<tr>
<td>Root reserved pages</td>
<td>-cr, -cR</td>
<td></td>
<td>-pr, -pR</td>
<td></td>
</tr>
<tr>
<td>Metadata for smart large objects</td>
<td>-cs, -cS</td>
<td></td>
<td>-ps, -pS</td>
<td></td>
</tr>
<tr>
<td>Space usage (by table or fragment)</td>
<td>CHECK PARTITION</td>
<td>PRINT PARTITION</td>
<td>-pt</td>
<td></td>
</tr>
<tr>
<td>Space usage (by table, with indexes)</td>
<td></td>
<td></td>
<td>-pT</td>
<td></td>
</tr>
</tbody>
</table>

### Using the -y Option to Perform Repairs

Use the -y option to instruct oncheck to perform repairs automatically, as the following examples show:

- `oncheck -cd -y`
- `oncheck -cD -y`
- `oncheck -ci -y`
- `oncheck -cI -y`

If you do not use the -y option, oncheck prompts you when it encounters an inconsistency and allows you to request a repair. If you specify option -n, oncheck does not prompt you because this option instructs oncheck to not perform repairs.

### Repairing Indexes in Sbspaces and External Spaces

The oncheck utility can repair an index in an sbspace or external space if the index is created using an access method that supports the oncheck -y option. Although the oncheck utility does not repair fragmented indexes, user-defined access methods can repair them. For more information about the oncheck options that access methods support, see the IBM Informix DataBlade API Programmer’s Guide or the IBM Informix Virtual-Index Interface Programmer’s Guide.

### Locking and oncheck

The oncheck utility places a shared lock on a table during the following operations, so no other users can perform updates, inserts, or deletes until the check has completed:

- When it checks data
- When it checks indexes (with -ci, -cI, -pk, -pK, -pl, or -pL) and the table uses page locking
- When you specify the -x option with -ci, -cI, -pk, -pK, -pl, or -pL and the table uses row locking
If the table does not use page locking, the database server does not place a shared lock on the table when you check an index with the `oncheck -ci,-cl,-pk,-pK,-pl, or -pL` options. When no shared lock is on the table during an index check, other users can update rows during the check.

By not placing a shared lock on tables using row locks during index checks, the `oncheck` utility cannot be as accurate in the index check. For absolute assurance of a complete index check, you can execute `oncheck` with the `-x` option. With the `-x` option, `oncheck` places a shared lock on the table, and no other users can perform updates, inserts, or deletes until the check has completed.

For more information about the `-x` option, refer to ["Turn On Locking with -x” on page 7-20](#). For information on shared locks and intent shared locks, see the IBM Informix Performance Guide.

The `oncheck` utility places a shared lock on system catalog tables when they are checked. It places an exclusive lock on a table when it executes.

### Oncheck Syntax

```
oncheck  
  -ce  
  -pe  
  -cr  
  -pr  
  -ck  
  -pk  
  -pl  
  -cd  
  -cl  
  -cc  
  -pc  
  -pb  
  -pt  
  -pd  
  -pp  
  -cs  
  -ps  
  -u  
  -option  
  -V  
  -version  
```

---

7-4 IBM Informix Dynamic Server Administrator’s Reference
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-cc</td>
<td>Checks system catalog tables for the specified database</td>
<td>See “oncheck -cc and -pc: Check system catalog tables” on page 7-8</td>
</tr>
<tr>
<td>-cd</td>
<td>Reads all pages except simple large objects from the tblspace for the specified database, table, or fragment and checks each page for consistency. Also checks tables that use a user-defined access method.</td>
<td>Does not check simple or smart large objects. See “oncheck -cd and -cD: Check pages” on page 7-9</td>
</tr>
<tr>
<td>-cD</td>
<td>Same as -cd but also reads the header of each blobpage and checks it for consistency.</td>
<td>Checks simple large objects but not smart large objects. See “oncheck -cd and -cD: Check pages” on page 7-9</td>
</tr>
<tr>
<td>-ce</td>
<td>Checks each chunk-free list and corresponding free space and each tblspace extent. Also checks smart-large-object extents and tblspace metadata.</td>
<td>The oncheck process verifies that the extents on disk correspond to the current control information that describes them. See “oncheck -ce, -pe: Check the chunk-free list” on page 7-10. For background information, see “Next-Extent Allocation” on page 4-11</td>
</tr>
<tr>
<td>-ci</td>
<td>Checks the ordering of key values and the consistency of horizontal and vertical node links for all indexes associated with the specified table. Also checks indexes that use a user-defined access method.</td>
<td>See “oncheck -ci and -cl: Check index node links” on page 7-10</td>
</tr>
<tr>
<td>-cl</td>
<td>Same as -ci but also checks that the key value tied to a rowid in an index is the same as the key value in the row.</td>
<td>See “oncheck -ci and -cl: Check index node links” on page 7-10</td>
</tr>
<tr>
<td>-cr</td>
<td>Checks each of the root dbspace reserved pages for several conditions.</td>
<td>See “oncheck -cr and -cR: Check reserved pages” on page 7-12</td>
</tr>
<tr>
<td>-cR</td>
<td>Checks the root dbspace reserved pages, physical-log pages, and logical-log pages.</td>
<td>See “oncheck -cr and -cR: Check reserved pages” on page 7-12</td>
</tr>
<tr>
<td>-cs</td>
<td>Checks smart large object and tblspace metadata for an tblspace.</td>
<td>See “oncheck -cs, -cS, -ps, -pS: Check and display tblspaces” on page 7-12</td>
</tr>
<tr>
<td>-cS</td>
<td>Checks smart large object and tblspace metadata for an tblspace as well as extents.</td>
<td>See “oncheck -cs, -cS, -ps, -pS: Check and display tblspaces” on page 7-12</td>
</tr>
</tbody>
</table>

**sbspace**
- Indicates optional sbspace name
- If not supplied, all sbspaces are checked.

**-n**
- Indicates that no index repair should be performed, even if errors are detected
- Use with the index repair options (-ci, -cl, -pk, -pK, -pI, and -pL).
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-pB</td>
<td>Displays statistics that describe the average fullness of blobspace blobpages in a specified table</td>
<td>These statistics provide a measure of storage efficiency for individual simple large objects in a database or table. If a table or fragment is not specified, statistics are displayed for the entire database. See “oncheck -pB: Display blobspace statistics” on page 7-13. For information about optimizing blobspace blobpage size, see the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-pc</td>
<td>Same as -ce but also displays the system catalog information as it checks the system catalog tables, including extent use for each table</td>
<td>None.</td>
</tr>
<tr>
<td>-pd</td>
<td>Displays rows in hexadecimal format</td>
<td>See “oncheck -pd and pD: Display rows in hexadecimal format” on page 7-13</td>
</tr>
<tr>
<td>-pD</td>
<td>Displays rows in hexadecimal format and simple-large-object values stored in the tblspace or header information for smart large objects stored in an sbspace sbpage and simple large objects stored in a blobspace blobpage</td>
<td>See “oncheck -pd and pD: Display rows in hexadecimal format” on page 7-13</td>
</tr>
<tr>
<td>-pe</td>
<td>Same as -ce but also displays the chunk and tblspace extent information as it checks the chunk free list, the corresponding free space, and each tblspace extent</td>
<td>See “oncheck -ce, -pe: Check the chunk-free list” on page 7-10</td>
</tr>
<tr>
<td>-pk</td>
<td>Same as -eI but also displays the key values for all indexes on the specified table as it checks them</td>
<td>See “oncheck -pk, -pK, -pl, -pL: Display index information” on page 7-13</td>
</tr>
<tr>
<td>-pK</td>
<td>Same as -eI but also displays the key values and rowids as it checks them</td>
<td>See “oncheck -pk, -pK, -pl, -pL: Display index information” on page 7-13</td>
</tr>
<tr>
<td>-pl</td>
<td>Same as -eI but also displays the key values. Only leaf-node index pages are checked</td>
<td>See “oncheck -pk, -pK, -pl, -pL: Display index information” on page 7-13</td>
</tr>
<tr>
<td>-pL</td>
<td>Same as -eI but also displays the key values and rowids for leaf-node index pages only</td>
<td>See “oncheck -pk, -pK, -pl, -pL: Display index information” on page 7-13</td>
</tr>
<tr>
<td>-pp</td>
<td>Displays contents of a logical page</td>
<td>See “oncheck -pp and -pP: Display the contents of a logical page” on page 7-16</td>
</tr>
<tr>
<td>-pP</td>
<td>Same as -pp but requires a chunk number and logical page number or internal rowid as input</td>
<td>See “oncheck -pp and -pP: Display the contents of a logical page” on page 7-16</td>
</tr>
<tr>
<td>-pr</td>
<td>Same as -eR but also displays the reserved-page information as it checks the reserved pages</td>
<td>See “oncheck -pr and pR: Display reserved-page information” on page 7-18</td>
</tr>
<tr>
<td>-pR</td>
<td>Same as -eR but also displays the information for the reserved pages, physical-log pages, and logical-log pages</td>
<td>See “oncheck -pr and pR: Display reserved-page information” on page 7-18</td>
</tr>
<tr>
<td>-ps</td>
<td>Checks and displays smart-large-object and sbspace metadata for an sbspace</td>
<td>See “oncheck -cs, -cS, -ps, -pS: Check and display sbspaces” on page 7-12</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-pS</td>
<td>Checks and displays smart-large-object and dbspace metadata. Lists extents and header information for individual smart large objects</td>
<td>See “oncheck -cs, -cS, -ps, -pS: Check and display dbspaces” on page 7-12</td>
</tr>
<tr>
<td>-pt</td>
<td>Displays tblspace information for a table or fragment</td>
<td>See “oncheck -pt and -pT: Display tblspaces for a table or fragment” on page 7-18</td>
</tr>
<tr>
<td>-pT</td>
<td>Same as -pt but also displays index-specific information and page-allocation information by page type (for dbspaces)</td>
<td>See “oncheck -pt and -pT: Display tblspaces for a table or fragment” on page 7-18</td>
</tr>
<tr>
<td>-q</td>
<td>Suppresses all checking and validation message</td>
<td>None</td>
</tr>
<tr>
<td>-x</td>
<td>Places a shared lock on the table when you check and print an index</td>
<td>Use with the -ci, -cI, -pk, -pK, -pl, or -pL options. For complete information, see “Turn On Locking with -x” on page 7-26</td>
</tr>
<tr>
<td>-y</td>
<td>Repairs indexes when errors are detected</td>
<td>None</td>
</tr>
<tr>
<td>-V</td>
<td>Displays the software version number and the serial number</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
<tr>
<td>-version</td>
<td>Displays the build version, host, OS, number and date, as well as the GLS version</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
<tr>
<td>chunknum</td>
<td>Specifies a decimal value that you use to indicate a particular chunk</td>
<td>Value must be an unsigned integer greater than 0. Chunk must exist.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Execute the -pe option to learn which chunk numbers are associated with specific dbspaces, blobspaces or dbspaces.</td>
</tr>
<tr>
<td>database</td>
<td>Specifies the name of a database that you want to check for consistency</td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>db1</td>
<td>Specifies the local database that contains a data type that you want to check</td>
<td>Optionally specify the local database server name using the format db1@server1.</td>
</tr>
<tr>
<td>db2</td>
<td>Specifies the remote database that contains a data type that you want to check</td>
<td>Optionally specify the remote database server name using the format db2@server2.</td>
</tr>
<tr>
<td>frag_dbs</td>
<td>Specifies the name of a dbspace that contains a fragment you want to check for consistency</td>
<td>Dbspace must exist and contain the fragment that you want to check for consistency. Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>index_name</td>
<td>Specifies the name of the index that you want to check for consistency</td>
<td>Index must exist on table and in database specified.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>logical_pagenum</td>
<td>Specifies an integer value that you use to indicate a particular page in a tblspace</td>
<td>Value must be an unsigned integer between 0 and 16,777,215, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.</td>
</tr>
<tr>
<td>object</td>
<td>Specifies the name of the DataBlade, cast, operator class, user-defined data type, or UDR that you want to check</td>
<td>If you do not specify an object name, the database server compares all objects of the same type with the same name and owner.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>--------------------</td>
</tr>
<tr>
<td>owner</td>
<td>Specifies the owner of a table</td>
<td>You must specify the current owner of the table. Syntax must conform to the Owner Name segment; for more information, see <em>IBM Informix Guide to SQL: Syntax</em>.</td>
</tr>
<tr>
<td>pagenum</td>
<td>Indicates the page number of the sbspace metadata portion to check and display</td>
<td>None.</td>
</tr>
<tr>
<td>partnum</td>
<td>Identifies the sbspace metadata partition to check and display</td>
<td>None.</td>
</tr>
<tr>
<td>rowid</td>
<td>Identifies the rowid of the row whose contents you want to display. The rowid is displayed as part of oncheck -pD output</td>
<td>Value must be an unsigned integer between 0 and 4,277,659,295, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.</td>
</tr>
<tr>
<td>sbspace</td>
<td>Specifies the name of the sbspace that you want to check for consistency</td>
<td>None.</td>
</tr>
<tr>
<td>server</td>
<td>Specifies the database server name</td>
<td>If you omit the database server name, oncheck uses the name that INFORMIXSERVER specifies.</td>
</tr>
<tr>
<td>table</td>
<td>Specifies the name of the table that you want to check for consistency</td>
<td>Table exists when you execute the utility. Syntax must conform to the Table Name segment; for more information, see <em>IBM Informix Guide to SQL: Syntax</em>.</td>
</tr>
<tr>
<td>tblspacenum</td>
<td>Identifies the tblspace whose contents you want to display</td>
<td>Value must be an unsigned integer between 0 and 208,666,624, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.</td>
</tr>
</tbody>
</table>

**oncheck -cc and -pc: Check system catalog tables**

**Syntax:**

```
oncheck -cc database
```

The -cc option checks all system catalog tables for the specified database. If you do not specify a database, it checks all system catalog tables for all databases.

The -pc option performs the same checks on system catalog tables and also displays the system catalog information, including the physical address, type of locking used, row size, number of keys, extent use, the number of pages allocated and used, tblspace partnum, and index use for each table.

Before you execute oncheck -cc or oncheck -pc, execute the SQL statement UPDATE STATISTICS to ensure that an accurate check occurs. To check a table, oncheck compares each system catalog table to its corresponding entry in the tblspace. For more information about the tblspace, see “Structure of the Tblspace Tblspace” on page 4-5"
oncheck -cd and-cD: Check pages

Syntax:

```
oncheck -cd database -owner table -frag_dbs -frag_part
```

The -cd option reads all pages, excluding blobpages and sbpages, from the tblspace for the specified database, table, fragment, or multiple fragments (fragparts), and checks each page for consistency. It checks entries in the bitmap page against the pages to verify mapping.

The -cD option performs the same checks as oncheck -cd, and also checks the header of each blobpage for consistency. The oncheck -cD option does not compare the beginning time stamps stored in the header with ending time stamps stored at the end of a blobpage. Use the oncheck -cD -y option to clean up orphaned simple large objects in blobspaces, which may occur after a rollback across several log files.

If the database contains fragmented tables, but no fragment is specified, oncheck -cd checks all fragments in the table. If you do not specify a table, it checks all tables in the database. By comparison, the -pd option displays a hexadecimal dump of specified pages but does not check for consistency.

For both the -cd and -cD options, the oncheck utility locks each table as it checks the indexes for the table. To repair the pages, specify oncheck -cd -y or oncheck -cD -y.

If tables are fragmented on multiple partitions in the same dbspace, the oncheck -cd and oncheck -cD commands show the partition names. The following example shows typical output for a table that has fragments in multiple partitions in the same dbspace:

```
TBLspace data check for multipart:informix.t1
  Table fragment partition part_1 in DBspace dbs1
  Table fragment partition part_2 in DBspace dbs1
  Table fragment partition part_3 in DBspace dbs1
  Table fragment partition part_4 in DBspace dbs1
  Table fragment partition part_5 in DBspace dbs1
```

The following example checks the data rows, including simple large objects and smart large objects, in the catalog table:

```
oncheck -cD superstores_demo:catalog
```

If oncheck finds an inconsistency, it displays a message similar to the following one:

```
BAD PAGE 2:28: pg_addr 2:28 != bp->bf_pagenum 2:69
```

The physical address 2:28 represents page 28 of chunk number 2. If oncheck finds no inconsistencies, it displays a header similar to the following one for each table that it checks:

```
TBLSPACE data check for stores_demo:informix.customer
```
If you specify a single fragment, **oncheck** displays a single header for that fragment. The **oncheck** utility displays a header similar to the following one for fragmented tables, one per fragment:

TBLspace data check for stores_demo:informix.tab1
  Table fragment in DBspace db1

If an index that uses an access method provided by a DataBlade module cannot find the access method, you receive the following message:

-9045 Access method access_method_name does not exist in database.
Ensure that the DataBlade installation was successful.

To monitor blobspace blobpages, refer to the **oncheck** -pB information at "oncheck -pB: Display blobspace statistics" on page 7-13.

Use CHECK DATA as the Administrative API command string for **oncheck** -cd.

**oncheck** -ce, -pe: Check the chunk-free list

**Syntax:**

```
$ oncheck -ce -pe
```

The -ce option checks each chunk-free list and corresponding free space and each tblspace extent. For more information, refer to "Next-Extent Allocation" on page 4-11 and "Structure of the Chunk Free-List Page" on page 4-4 respectively. The **oncheck** process verifies that the extents on disk correspond to the current control information that describes them.

The -pe option performs the same checks and also displays the chunk and tblspace extent information during the check. The -ce and -pe options also check blobspaces, smart-large-object extents, and user-data and metadata information in sbspace chunks.

For information about using **oncheck** -ce and -pe, see managing disk space in the *IBM Informix Administrator’s Guide*.

Use CHECK EXTENTS as the Administrative API command string for **oncheck** -ce.

**oncheck** -ci and -cl: Check index node links

**Syntax:**

```
$ oncheck -ci -cl
```

The -ci option checks the ordering of key values and the consistency of horizontal and vertical node links for all indexes associated with the specified table. For more information about indexes, see "Structure of B-Tree Index Pages" on page 4-16.

The -cl option performs the same checks as -ci and it also checks that the key value tied to a rowid in an index is the same as the key value in the row. The -cl option does not cross-check data on a functional index.
If you do not specify an index, the option checks all indexes. If you do not specify a table, the option checks all tables in the database.

The same -ci repair options are available with -cl. If oncheck -ci or oncheck -cl detects inconsistencies, it prompts you for confirmation to repair the problem index. If you specify the -y (yes) option, indexes are automatically repaired. If you specify the -n (no) option, the problem is reported but not repaired; no prompting occurs.

If oncheck does not find inconsistencies, the following message appears:
validating indexes......

The message displays the names of the indexes that oncheck is checking.

Note: Using oncheck to rebuild indexes can be time consuming. Processing is usually faster if you use the SQL statements DROP INDEX and CREATE INDEX to drop and re-create the index.

The following example checks all indexes on the customer table:
oncheck -cl -n stores_demo:customer

The following example checks the index zip_ix on the customer table:
oncheck -cl -n stores_demo:customer#zip_ix

If indexes are fragmented on multiple partitions in the same dbspace, the oncheck -ci and oncheck -cl commands show the partition names. The following example shows typical output for an index that has fragments in multiple partitions in the same dbspace:
Validating indexes for multipart:informix.t1...
  Index 'idx_t1'
    Index fragment partition part_1 in DBspace dbs1
    Index fragment partition part_2 in DBspace dbs1
    Index fragment partition part_3 in DBspace dbs1
    Index fragment partition part_4 in DBspace dbs1
    Index fragment partition part_5 in DBspace dbs1

By default, the database server does not place a shared lock on the table when you check an index with the oncheck -ci or -cl options unless the table uses page locking. For absolute assurance of a complete index check, you can execute oncheck -ci or oncheck -cl with the -x option. With the -x option, oncheck places a shared lock on the table, and no other users can perform updates, inserts, or deletes until the check has completed. For more information on using the -x option, [Turn On Locking with -x" on page 7-20]

When you execute oncheck on an external index, the user-defined access method is responsible for checking and repairing an index. If an index that employs a user-defined access method cannot find the access method, the database server reports an error. The oncheck utility does not repair inconsistencies in external indexes. You should not use oncheck -cl on a table that contains more than one type of index.

Important: If you are using the Verity Text Search DataBlade Module, the -cl option performs an index merge instead of the usual operations.
oncheck -cr and -cR: Check reserved pages

Syntax:

```
oncheck -cr
```

The -cr option checks each of the root dbspace reserved pages as follows:

- It validates the contents of the ONCONFIG file with the PAGE_CONFIG reserved page.
- It ensures that all chunks can be opened, that chunks do not overlap, and that chunk sizes are correct.

The -cR option performs the same checking and validation, and also checks all logical-log and physical-log pages for consistency. The -cr option is considerably faster because it does not check the log-file pages.

If you have changed the value of a configuration parameter (either through ISA, onparams, onmonitor, onspaces, or by editing the configuration file), but you have not yet reinitialized shared memory, oncheck -cr and oncheck -cR detect the inconsistency and return an error message.

If oncheck -cr does not display any error messages after you execute it, you can assume that all three items in the preceding list were checked successfully.

For more information on reserved pages, see "Reserved Pages" on page 4-3.

oncheck -cs, -cS, -ps, -pS: Check and display sbspaces

Syntax:

```
oncheck -cs
```

The -cs option checks sbspaces. The -ps option checks sbspaces and extents.

The -cS option validates and displays metadata for an sbspace.

The -ps option checks sbspaces and extents. If you do not specify the sbspace name, these options check all sbspaces.

The -pS option validates and displays metadata for an sbspace and also lists extents and header information for smart large objects.

If you do not specify the sbspace name, all sbspaces will be checked. The following example checks and displays metadata for test_sbspace:

```
oncheck -ps test_sbspace
```

If you specify rootdbs as the sbspace name with the -cs or -ps options, oncheck checks the root dbspace.
For more information about using the -cs, -cS, -ps, and -pS options, see the IBM Informix Administrator’s Guide.

**oncheck -pB: Display blobspace statistics**

**Syntax:**

```
:oncheck -pB database:
   - owner.
   - table
```

The -pB option displays statistics that describe the average fullness of blobspace blobpages in a specified table. These statistics provide a measure of storage efficiency for individual simple large objects in a database or table. If you do not specify a table or fragment, the option displays statistics for the entire database. For more information, see optimizing blobspace blobpage size in the chapter on managing disk space in the IBM Informix Administrator’s Guide.

**oncheck -pd and pD: Display rows in hexadecimal format**

**Syntax:**

```
:oncheck -pd database:
   - owner.
   - table
   - frag_dbs
   - frag_part
   - rowid
   - tblspacenum
   - logical_pagenum
```

The -pd option takes a database, a table, a fragment, a fragment partition (fragpart), and a specific rowid or tblspace number and logical page number as input. In every case, -pd prints page-header information and displays the specified rows for the database object (database, table, fragment, internal rowid, or page number) that you specify in hexadecimal and ASCII format. No checks for consistency are performed.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>database</td>
<td>Specifies the name of a database that you want to check for consistency</td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>frag_dbs</td>
<td>Specifies the name of a dbspace that contains a fragment you want to check for consistency</td>
<td>Dbspace must exist and contain the fragment that you want to check for consistency. Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>frag_part</td>
<td>Specifies the fragment partition</td>
<td>For fragmented tables or an index that use expression-based or round-robin distribution schemes, you can create multiple partitions, which are collections of pages for a table or index, within a single dbspace. This partition is referred to as a fragment partition or fragpart.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>logical page num</td>
<td>Specifies an integer value that you use to indicate a particular page in a tblspace</td>
<td>Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier. Value must be an unsigned integer between 0 and 16,777,215, inclusive.</td>
</tr>
<tr>
<td>owner</td>
<td>Specifies the owner of a table</td>
<td>You must specify the current owner of the table. Syntax must conform to the Owner Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>rowid</td>
<td>Identifies the rowid of the row whose contents you want to display. The rowid is displayed as part of oncheck -pD output</td>
<td>Value must be an unsigned integer between 0 and 4,277,659,295, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.</td>
</tr>
<tr>
<td>table</td>
<td>Specifies the name of the table that you want to check for consistency</td>
<td>Table exists when you execute the utility. Syntax must conform to the Table Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>tblspace num</td>
<td>Identifies the tblspace whose contents you want to display</td>
<td>Value must be an unsigned integer between 0 and 208,666,624, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.</td>
</tr>
</tbody>
</table>

If you specify an internal rowid (expressed as a hexadecimal value), the rowid maps to a particular page, and all rows from that page are printed.

If you specify a logical page number (expressed as a decimal), all the rows of the tblspace number with the logical page number are printed.

If you specify a fragment, all the rows in the fragment are printed, with their rowids, forward pointers, and page type.

If you specify a table, all the rows in the table are printed, with their rowids, forward pointers, and page type.

If you specify a database, all the rows in all the tables in the database are printed. TEXT and BYTE column descriptors stored in the data row are printed, but TEXT and BYTE data itself is not.

The -pD option prints the same information as -pd. In addition, -pD prints TEXT and BYTE values stored in the tblspace or header information for simple large objects stored in a blobspace blobpage. The following example show different options for the oncheck -pd and oncheck -pD commands:

```bash
oncheck -pd stores_demo:customer,frgmt1
oncheck -pd stores_demo:customer
oncheck -pD stores_demo:customer 0x101
```

The following example shows a partial output of an oncheck -pD command:

```bash
oncheck -pD multipart:t1:

TBLspace data check for multipart:informix.t1
    Table fragment partition part_1 in DBspace dbs1
page_type rowid length fwd_ptr
HOME 101 24 0
```

7-14 IBM Informix Dynamic Server Administrator’s Reference
oncheck -pk, -pK, -pl, -pL: Display index information

Syntax:

```
oncheck -pk database -pK table -pl owner -pL
```

The **-pk** option performs the same checks as the **-ci** option and in addition, displays the key values for all indexes on the specified table as it checks them.

The **-pK** option performs the same checks as the **-ci** option and in addition, displays the key values and rowids as it checks them.

The **-pl** option performs the same checks as the **-ci** option and displays the key values, but checks only leaf-node index pages. It ignores the root and branch-node pages.

The **-pL** option performs the same checks as the **-ci** option and displays the key values and rowids, but checks only leaf-node index pages. It ignores the root and branch-node pages.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>database</strong></td>
<td>Specifies the name of a database that you want to check for consistency</td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td><strong>index_name</strong></td>
<td>Specifies the name of the index that you want to check for consistency</td>
<td>Index must exist on table and in database specified. Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td><strong>owner</strong></td>
<td>Specifies the owner of a table</td>
<td>You must specify the current owner of the table. Syntax must conform to the Owner Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td><strong>table</strong></td>
<td>Specifies the name of the table that you want to check for consistency</td>
<td>Table exists when you execute the utility. Syntax must conform to the Table Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td><strong>-x</strong></td>
<td>Places a shared lock on the table when you check and print an index</td>
<td>For complete information, see &quot;Turn On Locking with -x&quot; on page 7-20</td>
</tr>
</tbody>
</table>

If any of the oncheck options detect inconsistencies, you are prompted for confirmation to repair the problem index. If you specify the **-y** (yes) option, indexes are automatically repaired. If you specify the **-n** (no) option, the problem is reported but not repaired; no prompting occurs.

The following example displays information about all indexes on the **customer** table:

```
oncheck -pl -n stores_demo:customer
```
The following example displays information about the index zip_ix, which was created on the customer table:

```bash
oncheck -pl -n stores_demo:customer#zip_ix
```

By default, the database server does not place a shared lock on the table when you check an index with the `oncheck -pk`, `oncheck -pl`, or `oncheck -pl` options unless the table uses page locking. For absolute assurance of a complete index check, you can execute `oncheck -pk, oncheck -pl, oncheck -pl` or `oncheck -pl` with the `-x` option. With the `-x` option, `oncheck` places a shared lock on the table, and no other users can perform updates, inserts, or deletes until the check has completed. For more information on using the `-x` option, see "Turn On Locking with -x" on page 7-20.

For more information on `oncheck -ci`, see "oncheck -ci and -cl: Check index node links" on page 7-10. For more information on index pages, see "Structure of B-Tree Index Pages" on page 4-16.

---

**oncheck -pp and -PP: Display the contents of a logical page**

**Syntax:**

```bash
oncheck -pp [database:owner.table,frag_dbs,frag_part,rowid] -pl
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>database</td>
<td>Specifies the name of a database that you want to check for consistency</td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>chunknum</td>
<td>Specifies a decimal value that you use to indicate a particular chunk</td>
<td>Value must be an unsigned integer greater than 0. Chunk must exist.</td>
</tr>
<tr>
<td>frag_dbs</td>
<td>Specifies the name of a dbspace that contains a fragment you want to check for consistency</td>
<td>Dbspace must exist and contain the fragment that you want to check for consistency. Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>frag_part</td>
<td>Specifies the fragpart.</td>
<td>For fragmented tables or an index that use expression-based or round-robin distribution schemes, you can create multiple partitions, which are collections of pages for a table or index, within a single dbspace. This partition is referred to as a fragment partition or fragpart.</td>
</tr>
<tr>
<td>logical_pagenum</td>
<td>Specifies an integer value that you use to indicate a particular page in a tblspace</td>
<td>Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier. Value must be an unsigned integer between 0 and 16,777,215, inclusive.</td>
</tr>
<tr>
<td>owner</td>
<td>Specifies the owner of a table</td>
<td>You must specify the current owner of the table. Syntax must conform to the Owner Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>rowid</td>
<td>Identifies the rowid of the row whose contents you want to display</td>
<td>Value must be an unsigned integer between 0 and 4,277,659,295, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>table</td>
<td>Specifies the name of the table that you want to check for consistency</td>
<td>Table exists when you execute the utility.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Syntax must conform to the Table Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>tblspacenum</td>
<td>Identities the tblspace whose contents you want to display</td>
<td>Value must be an unsigned integer between 0 and 208,666,624, inclusive.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.</td>
</tr>
</tbody>
</table>

The **-pp** option has the following syntax variations:

<table>
<thead>
<tr>
<th>Invocation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>oncheck -pp tblspc lpn &lt;pages&gt;</td>
<td>Displays the contents of a logical page using a tablespace number and logical page number. You can also specify an optional parameter specifying the number of pages to be printed.</td>
</tr>
<tr>
<td>oncheck -pp tblspc lpn -h</td>
<td>Displays only the header of a logical page using a tablespace number and logical page number.</td>
</tr>
<tr>
<td>oncheck -pp database:table rowid</td>
<td>Displays the contents of a logical page using a database name, table name, and an Informix internal rowid. You can obtain this internal rowid with the oncheck -pD command. This internal rowid is not the serial rowid that is assigned in tables created with the CREATE TABLE tabname WITH ROWIDS statement. For more information, see “Definition of Rowid” on page 4-13.</td>
</tr>
</tbody>
</table>

The page contents appear in ASCII format. The display also includes the number of slot-table entries on the page. The following example shows different invocations of the oncheck **-pp** command:

```bash
oncheck -pp stores_demo:orders 0x211 # database:owner.table, # fragment rowid
oncheck -pp stores_demo:informix.customer,frag_dbspce1 0x211
oncheck -pp 0x100000a 25 # specify the tblspace number and # logical page number
```

The **-pP** option provides the following syntax variations:

<table>
<thead>
<tr>
<th>Invocation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>oncheck -pP chunk# offset pages</td>
<td>Displays the contents of a logical page using a chunk number and an offset. You can also specify an optional parameter specifying the number of pages to be printed.</td>
</tr>
<tr>
<td>oncheck -pP chunk# offset -h</td>
<td>Displays only the header of a logical page using a chunk number and an offset.</td>
</tr>
</tbody>
</table>

**Note:** The output for chunk page displays both the start and the length fields in decimal format.

The following example shows typical output using the onstat **-pP** command:

```bash
oncheck -pP 1 5 2
addr  stamp  slott  ptr  len  addr  stamp  nslots  flag  type  frptr  frcnt  next  prev  stamp
100005  250181  2  1000  ROOTRSV  320  1716  0  0  250181
```

Chapter 7. The oncheck Utility  7-17
oncheck -pr and pR: Display reserved-page information

Syntax:

```
/oncheck /pr
```

The -pr option performs the same checks as oncheck -cr and displays the reserved-page information.

The -pR option performs the same checks as oncheck -cR, displays the reserved-page information, and displays detailed information about logical-log and physical-log pages, marking the start and end of the active physical-log pages.

If you have changed the value of a configuration parameter (either through ISA or by editing the configuration file), but you have not yet reinitialized shared memory, oncheck -pr and oncheck -pR detect the inconsistency and return an error message.

For a listing and explanation of oncheck -pr output, see "Reserved Pages" on page 4-3. For a description of the -cr option, see "oncheck -cr and -cR: Check reserved pages" on page 7-12.

oncheck -pt and -pT: Display tblspaces for a table or fragment

Syntax:

```
/oncheck /pt /database /owner/.table/.frag_dbs
```

The -pt option prints a tblspace report for a given table or fragment whose name and database you specify when you execute oncheck at the command line. The report contains general allocation information including the maximum row size, the number of keys, the number of extents, their sizes, the pages allocated and used per extent, the current serial value, and the date that the table was created. The -pt output prints the pagesize of the tblspace, the number of pages (allocated, used and data) in terms of logical pages. The Extents fields list the physical address for the tblspace tblspace entry for the table and the address of the first page of the first extent. The extent list shows the number of logical as well as physical pages in every extent. If you do not specify a table, the option displays this information for all tables in the database.

The -pT option prints the same information as the -pt option. In addition, the -pT option displays index-specific information and page-allocation information by page type (for dbspaces).
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>database</td>
<td>Specifies the name of a database that you want to check for consistency</td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>frag_dbs</td>
<td>Specifies the name of a dbspace that contains a fragment you want to check for consistency</td>
<td>Dbspace must exist and contain the fragment that you want to check for consistency. Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>owner</td>
<td>Specifies the owner of a table</td>
<td>You must specify the current owner of the table.                                   Syntax must conform to the Owner Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>table</td>
<td>Specifies the name of the table that you want to check for consistency</td>
<td>Table exists when you execute the utility.                                          Syntax must conform to the Table Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
</tbody>
</table>

Output for both -pt and -pT contains listings for **Number of pages used**. The value shown in the output for this field is never decremented because the disk space allocated to a tblspace as part of an extent remains dedicated to that extent even after you free space by deleting rows. For an accurate count of the number of pages currently used, refer to the detailed information on tblspace use (organized by page type) that the -pT option provides.

The following example shows an example of output of the oncheck -pt command:

```
TBLspace Report for testdb:tab1

  Physical Address    2:10
  Creation date       10/07/2004 17:01:16
  TBLspace Flags      801 Page Locking
                      TBLspace use 4 bit bit-maps

  Maximum row size    14
  Number of special columns 0
  Number of keys      0
  Number of extents   1
  Current serial value 1
  Pagesize (k)        4
  First extent size   4
  Next extent size    4
  Number of pages allocated 340
  Number of pages used 337
  Number of data pages 336
  Number of rows      75806
  Partition partnum   2097154
  Partition lockid    2097154

  Extents
  Logical Page | Physical Page | Size Physical Pages |
                | 0            | 2:106             | 340   | 680   |

```

For more examples of using oncheck -pt and -pT, see managing disk space in the IBM Informix Administrator's Guide and the IBM Informix Performance Guide.
## Turn On Locking with -x

The -x option can be appended to the -ci, -cI, -pk, -pK, -pl, and -pL options to place a shared lock on affected tables. While the table is locked, no other users can perform inserts, updates, and deletions while oncheck checks or prints the index. Without the -x option for tables with row locking, oncheck only places an IS (intent shared) lock on the table, which prevents actions such as dropping the table or the indexes during the check.

For example, the following sample command instructs oncheck to lock indexes for the customer table while it validates the order of key values, validates horizontal links, and ensures that no node appears twice in the index:

```
oncheck -cix stores_demo:customer```

When you specify option -x, oncheck locks indexes for tables that use row locking. If oncheck detects page-lock mode, it displays a warning message and places a shared lock on the table regardless.

## Send Special Arguments to the Access Method with -u

You can use the -u option to send special arguments to the access method. The possible arguments depend on the access method. For example, the R-tree access method supports the display option, as the following example shows:

```
oncheck -pl -u "display"```

Use commas to separate multiple arguments in the argument string.

For information on valid arguments for your access method, refer to the user manual for your access method.

## Return Codes on Exit

The oncheck utility returns the following codes on exit.

GLS failures:-1
Invalid serial/key:2
Onconfig access error:2
Invalid onconfig settings:2
Invalid arguments to oncheck:2
Error connecting database server:1
error detected by oncheck:2
no errors detected by oncheck:0

Windows only:
Not properly installed:1
Authentication error:2
Chapter 8. The ondblog Utility

In This Chapter

This chapter shows you how to use the ondblog utility.

ondblog: Change Logging Mode

The ondblog utility lets you change the logging mode for one or more databases. Alternatively, you can change the logging mode by using the ALTER LOGMODE Administrative API command string. For information on using Administrative API commands, see IBM Informix Guide to SQL: Syntax.

The ondblog utility logs its output in the BAR_ACT_LOG file.

If you turn on transaction logging for a database, you must create a level-0 backup of all of the storage spaces that contain data in the database before the change takes effect.

For more information and examples of logging modes, see the following topics in the chapter on managing database-logging status in the IBM Informix Administrator's Guide:
- Modifying the database-logging status
- Modifying table-logging status

ondblog Syntax

```
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>buf</td>
<td>Sets the logging mode so that transaction information is written to a buffer before it is written to a logical log</td>
<td>None.</td>
</tr>
<tr>
<td>unbuf</td>
<td>Sets the logging mode so that data is not written to a buffer before it is written to a logical log</td>
<td>None.</td>
</tr>
<tr>
<td>nolog</td>
<td>Sets the logging mode so that no database transactions are logged</td>
<td>None.</td>
</tr>
<tr>
<td>ansi</td>
<td>Changes database logging to be ANSI compliant</td>
<td>Once you create or convert a database to ANSI mode, you cannot change it back to any of the other logging modes.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------</td>
</tr>
<tr>
<td>cancel</td>
<td>Cancels the logging-mode change request before the next level-0 backup occurs</td>
<td>None.</td>
</tr>
<tr>
<td>-f dbfile</td>
<td>Changes the logging status of the databases that are listed (one per line) in the text file whose pathname is given by dbfile</td>
<td>This command is useful if the list of databases is long or used often.</td>
</tr>
<tr>
<td>db_list</td>
<td>Names a space-delimited list of databases whose logging status is to be changed</td>
<td>If you do not specify anything, all databases that the database server manages are modified.</td>
</tr>
</tbody>
</table>
Chapter 9. The oninit Utility

oninit: Initialize the Database Server ................................................................. 9-1
oninit Syntax ........................................................................................................... 9-1
Initialize Shared Memory Only ................................................................................. 9-2
  Initializing Shared Memory with No Options ...................................................... 9-3
  Initializing Shared Memory with the -s Option .................................................. 9-3
Initialize Disk Space and Shared Memory ............................................................... 9-3
Specify the Number of Virtual Processors ............................................................ 9-3

oninit: Initialize the Database Server

Execute the oninit utility from the command line to initialize database server shared memory and bring the database server online. If you use the oninit -i option, you can also initialize disk space.

To initialize the database server in administration mode, use the oninit -j option. This is an administrator-only mode you can use to perform maintenance operations including those that require executing SQL or DDL commands. You can use the -j flag with other oninit flags, except the -s flag. When in administration mode, the system will only accept connection requests from the informix user. The server makes an entry in the online log whenever it enters or exits administration mode.

On UNIX, you must be logged in as user root or informix to execute oninit. User informix should be the only member of the group informix. On Windows, you must be a member of the Informix-Admin group.

Note: The oninit utility does not have a functionally equivalent Administrative API command string. You cannot initialize the database server through an Administrative API command string.

Before you initialize the database server, set the INFORMIXSERVER environment variable to the dbservername that you chose when you set the configuration parameter DBSERVERNAME. INFORMIXSERVER is not required for initialization. However, if INFORMIXSERVER is not set, the database server does not build the sysmaster tables. Also, the DB–Access utility requires INFORMIXSERVER to be set.

For information about what happens during initialization, see the chapter on initializing the database server in the IBM Informix Administrator’s Guide.

oninit Syntax

```
```

(1) Initialize Shared Memory Only
(2) Initialize Disk Space and Shared Memory

© Copyright IBM Corp. 1996, 2007 9-1
Notes:
1 see 9-2
2 see 9-3

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-y</td>
<td>Causes the database server to automatically respond yes to all prompts</td>
<td>None.</td>
</tr>
<tr>
<td>-j</td>
<td>Initializes the server in administration mode</td>
<td>The -j flag may be combined with other oninit flags, except the quiescent mode (-s) flag.</td>
</tr>
</tbody>
</table>

Note: If you enabled the FAST_RESTART_PHYSLOG parameter by setting it to 1 and the database server shuts down, you can initiate fast recovery by executing oninit without any options.

Initialize Shared Memory Only

Syntax:

```
-p -s -S -U
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p</td>
<td>Directs oninit not to search for (and delete) temporary tables</td>
<td>If you use this option, the database server returns to online mode more rapidly, but space used by temporary tables left on disk is not reclaimed.</td>
</tr>
<tr>
<td>-s</td>
<td>Initializes shared memory and leaves the database server in quiescent mode See “Initializing Shared Memory with the -s Option” on page 9-3.</td>
<td>The database server should be in offline mode to initialize shared memory. Do not use this flag in combination with the -j flag. Specifying both -j and -s will result in an error.</td>
</tr>
<tr>
<td>-S</td>
<td>Starts database server in standard mode; disables HDR</td>
<td>If you use the -S option, the database server starts as a standard server instead of as a primary or as a secondary HDR server. It will leave the database server in quiescent mode and will require a subsequent onmode -m command for multiuser access.</td>
</tr>
<tr>
<td>-U</td>
<td>Starts the database server and specifies a list of users who can access the server in Administration Mode.</td>
<td>You must specify comma-separated user names, such as: Karin,Sarah,Andrew. In addition to the users specified, the DBSA group and user informix can connect to the database server when it is in administration mode. Users specified in oninit -U are valid until the server instance is terminated or the onmode -j -U &quot; &quot; command is executed. This option overrides any users listed in the ONCONFIG file. To specify users in the ONCONFIG file, see “ADMIN_MODE_USERS” on page 1-13</td>
</tr>
</tbody>
</table>
Initializing Shared Memory with No Options
If you execute `oninit` without options, the database server is left in online mode after shared memory is initialized. For example, the following commands take the database server offline and back online:

```
onmode -ky
oninit
```

Initializing Shared Memory with the -s Option
The -s option initializes shared memory and leaves the database server in quiescent mode.

The following commands shut down and restart the database server in quiescent mode:

```
onmode -ky
oninit -s
```

Initialize Disk Space and Shared Memory

Syntax:

```
-i [-s]
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i</td>
<td>Causes the database server to initialize disk space and shared memory. Leaves the database server in online mode after it initializes disk space.</td>
<td>None.</td>
</tr>
<tr>
<td>-s</td>
<td>When used with -i, causes the database server to be left in quiescent mode after disk initialization.</td>
<td>None.</td>
</tr>
</tbody>
</table>

When Dynamic Server 10.0 or later is first initialized with the `oninit -iyv` command, by default it comes online with large chunk mode fully enabled. Reversion is not possible. For more information about allowing large chunk mode, see "onmode -BC: Allow large chunk mode" on page 11-4.

Warning: When you initialize disk space, the initialization destroys all data that your database server currently manages.

The database server must be offline when you initialize disk space.

Specify the Number of Virtual Processors
Use `VPCLASS cpu,num` and `VPCLASS aio` to specify the initial number of VPs for the CPU and AIO classes. For more information, see "VPCLASS" on page 1-103.

The VPCLASS configuration parameter allows you to specify, for each class of virtual processors, the number of VPs that the database server should start on initialization. Alternatively, you can use `NUMCPUVPS` and `NUMAIOVPS` to specify the initial number of VPs for the CPU and AIO classes. However, you cannot use both `VPCLASS` and `NUMCPUVPS` and `NUMAIOVPS` in the same configuration file. If your ONCONFIG file contains conflicting parameters, `oninit` returns one of the following messages:
oninit: Can't mix VPCLASS cpu and NUMCPUVPS, SINGLE_CPU_VP, AFF_SPROC, AFF_NPROCS, or NOAGE parameters

oninit: Can't mix VPCLASS aio and NUMAIOVPS parameters

For more information, refer to “VPCLASS” on page 1-103
Chapter 10. The onlog Utility

onlog: Display Logical-Log Contents .................................................. 10-1
onlog Syntax ................................................................. 10-1
Read Filters ................................................................. 10-2
Display Filters ............................................................... 10-2
If an Error Is Detected ...................................................... 10-2
Log-Record Read Filters ...................................................... 10-2
Log-Record Display Filters .................................................... 10-3

onlog: Display Logical-Log Contents

The onlog utility displays the contents of a logical-log file, either on disk or on backup.

The onlog output is useful in debugging situations when you want to track a specific transaction or see what changes have been made to a specific tblspace. (For information about interpreting the logical-log file contents, see Chapter 5, "Interpreting Logical-Log Records," on page 5-1)

Any user can run all of the onlog options except the -l option. Only user informix on UNIX or a member of the Informix-Admin group on Windows can run the -l option.

If the database server is in offline mode when you execute onlog, only the files on disk are read. If the database server is in quiescent or online mode, onlog also reads the logical-log records stored in the logical-log buffers in shared memory (after all records on disk have been read).

When the database server reads a logical-log file with status U from disk while in online mode, the database server denies all access to the logical-log files, effectively stopping database activity for all sessions. (For more information, see "onstat -l
Print physical and logical log information" on page 15-122.) For this reason, it is recommended that you wait until the files have been backed up and then read the contents of the logical-log files from backup.

Note: The onlog utility does not have a functionally equivalent Administrative API command string.

onlog Syntax

```
$ onlog

$ onlog [options] (1) Log-Record Read Filters [options] (2) Log-Record Display Filters

$ onlog [options] (1) Log-Record Read Filters [options] (2) Log-Record Display Filters

$ onlog [options] (1) Log-Record Read Filters [options] (2) Log-Record Display Filters

$ onlog [options] (1) Log-Record Read Filters [options] (2) Log-Record Display Filters

```
### Notes:
1. see 10-2
2. see 10-3

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>q</code></td>
<td>Suppresses the initial header and the one-line header that appears every 18 records by default</td>
<td>None.</td>
</tr>
<tr>
<td><code>-V</code></td>
<td>Displays the software version number and the serial number</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
<tr>
<td><code>-version</code></td>
<td>Displays the build version, host, OS, number and date, as well as the GLS version</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
</tbody>
</table>

### Read Filters
You direct **onlog** to read the following portions of the logical log as it searches for records to display:

- Records stored on disk
- Records stored on backup media
- Records from the specified logical-log file

By default, **onlog** displays the logical-log record header, which describes the transaction number and the record type. The record type identifies the type of operation performed.

In addition to the header, you can use the read filters to direct **onlog** to display the following information:

- Logical-log record header and data (including copies of simple large objects stored in a dbspace or tblspace)
- Copies of blobpages from blobspaces
  They are copied from the logical-log backup only. They are not available from disk.

### Display Filters
You can display every logical-log record header, or you can specify output based on the following criteria:

- Records associated with a specific table
- Records initiated by a specific user
- Records associated with a specific transaction

### If an Error Is Detected
If **onlog** detects an error in the log file, such as an unrecognizable log type, it displays the entire log page in hexadecimal format and terminates.

### Log-Record Read Filters
The **onlog** utility uses the pathnames that are stored in the root dbspace reserved pages to locate the logical-log files. If you use ON–Bar to back up the logical logs, **onlog** asks the storage manager to retrieve the appropriate logical-log records from the backup media.
Syntax:

-d device
-b [SV040000]
-n starting uniqid-ending uniqid

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-b</td>
<td>Displays logical-log records associated with blobspace blobpages</td>
<td>The database server stores these records on the logical-log backup media as part of blobspace logging.</td>
</tr>
<tr>
<td>-d device</td>
<td>Names the pathname of the storage device where the desired logical-log backup is mounted</td>
<td>If you use ontape, the device that you name must be the same as the pathname of the device assigned to the configuration parameter LTAPEDEV. If the -d option is not used, onlog reads the logical-log files stored on disk, starting with the logical-log file with the lowest logid. If you use ON-Bar to back up logical logs, use the onbar -P command to view the contents of a logical-log file. See the IBM Informix Backup and Restore Guide. For pathname syntax, see your operating-system documentation.</td>
</tr>
<tr>
<td>-n starting uniqid-ending uniqid</td>
<td>Directs onlog to read all the logical-log records contained in the log file that you specified from starting uniqid to the ending uniqid.</td>
<td>The starting uniqid and the ending uniqid are the unique ID numbers of the logical log. To determine the uniqid of a particular logical-log file, use the onstat -l command. If you do not use the -n option, onlog reads all the logical-log files that are available (either on disk or on tape). For information about the onstat utility, see “Monitor the Database Server Status” on page 15-4.</td>
</tr>
</tbody>
</table>

Log-Record Display Filters

Syntax:

-1
-t tblspace_num
-u username
-x transaction_id

Notes:
1 Only one occurrence of this item allowed
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-l</td>
<td>Displays the long listing of the logical-log record.</td>
<td>The long listing of a log record includes a complex hexadecimal and ASCII dump of the entire log record. The listing is not intended for casual use.</td>
</tr>
<tr>
<td>-t tblspace_num</td>
<td>Displays records associated with the tblspace that you specify.</td>
<td>Unsigned integer. Number, greater than 0, must be in the partnum column of the systables system catalog table. Specify this value as either an integer or hexadecimal value. (If you do not use a 0x prefix, the value is interpreted as an integer.) To determine the tblspace number of a particular tblspace, query the systables system catalog table as described in “Tblspace Numbers” on page 4-6.</td>
</tr>
<tr>
<td>-u username</td>
<td>Displays records for a specific user.</td>
<td>User name must be an existing login name. User name must conform to operating-system-specific rules for login name.</td>
</tr>
<tr>
<td>-x transaction_id</td>
<td>Displays only records associated with the transaction that you specify.</td>
<td>Value must be an unsigned integer between 0 and TRANSACTIONS - 1, inclusive. You should need to use the -x option only in the unlikely case that an error is generated during a rollforward. When this situation occurs, the database server sends a message to the message log that includes the transaction ID of the offending transaction. You can use this transaction ID with the -x option of onlog to investigate the cause of the error.</td>
</tr>
</tbody>
</table>

If you do not specify any options, onlog displays a short listing of all the records in the log. You can combine options with any other options to produce more selective filters. For example, if you use both the -u and -x options, onlog displays only the activities that the specified user initiated during the specified transaction. If you use both the -u and -t options, onlog displays only the activities initiated by the specified user and associated with the specified tblspace.
Chapter 11. The onmode Utility

In This Chapter
- onmode Syntax .......................................................... 11-3
- onmode -a: Add a shared-memory segment ..................... 11-4
- onmode -BC: Allow large chunk mode ............................ 11-4
- onmode -C: Force a checkpoint ..................................... 11-5
- onmode -C: Control the B-tree scanner ......................... 11-6
- onmode -d: Set data-replication types ............................ 11-7
  - Using the -d standard Option .................................. 11-7
  - Using the -d primary dbservename Option .................... 11-7
  - Using the -d secondary dbservename Option ................ 11-7
- onmode -d: Set secondary server characteristics ............... 11-8
- onmode -d: Replicate an index with data-replication .......... 11-9
- onmode -D, -M, -Q, -S: Change decision-support parameters 11-10
- onmode -e: Change usage of the SQL statement cache .......... 11-11
- onmode -F: Free unused memory segments ..................... 11-12
- onmode -k, -m, -s, -u, -j: Change database server mode ... 11-12
  - Taking the Database Server to Offline Mode with the -k Option 11-13
  - Bringing the Database Server Online with the -m Option 11-14
  - Shutting Down the Database Server Gracefully with the -s Option 11-14
  - Shutting Down the Database Server Immediately with the -u Option 11-14
  - Changing the Database Server to Administration Mode with the -j Option 11-14
  - Changing Database Server Mode with ON-Monitor (UNIX) 11-15
- onmode -i: Switch the logical-log file .......................... 11-15
- onmode -n, -r: Change shared-memory residency ............... 11-15
- onmode -O: Override ONDBSPACE=DOWN WAIT mode ........... 11-16
- onmode -p: Add or remove virtual processors .................. 11-17
  - Adding and Dropping Virtual Processors .................... 11-18
  - Dropping Virtual Processors Automatically ............... 11-18
  - Monitoring Poll Threads with onstat ....................... 11-19
- onmode -R: Regenerate .infos File ............................... 11-19
- onmode -W: Change settings for the SQL statement cache .... 11-20
  - SQL Statement Cache Examples ............................... 11-20
- onmode -wf, -wm: Dynamically change certain configuration parameters 11-20
- onmode -wm: Change LRU tuning status ......................... 11-21
- onmode -Y: Dynamically change SET EXPLAIN .................. 11-22
- onmode -z: Kill a database server session ..................... 11-23
- onmode -Z: Kill a distributed transaction ..................... 11-23

In This Chapter
This chapter shows you how to use the following onmode options:
- "onmode -a: Add a shared-memory segment” on page 11-4
- "onmode -BC: Allow large chunk mode” on page 11-4
- "onmode -C: Force a checkpoint” on page 11-5
- "onmode -C: Control the B-tree scanner” on page 11-6
- "onmode -d: Set data-replication types” on page 11-7
- "onmode -d: Set secondary server characteristics” on page 11-8
- "onmode -d: Replicate an index with data-replication” on page 11-9
- "onmode -D, -M, -Q, -S: Change decision-support parameters” on page 11-10
- "onmode -e: Change usage of the SQL statement cache” on page 11-11
- "onmode -F: Free unused memory segments” on page 11-12
If you do not use any options, the database server returns a usage statement.

On UNIX, you must be user root or user informix to execute onmode. On Windows, you must be a member of the Informix-Admin group.

The following onmode options have equivalent Administrative API command strings:

<table>
<thead>
<tr>
<th>onmode option</th>
<th>Administrative API command string</th>
</tr>
</thead>
<tbody>
<tr>
<td>onmode -a</td>
<td>ADD MEMORY</td>
</tr>
<tr>
<td>onmode -a seg_size</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -BC [1</td>
<td>2]</td>
</tr>
<tr>
<td>onmode -c</td>
<td>CHECKPOINT</td>
</tr>
<tr>
<td>onmode -c { block</td>
<td>unblock }</td>
</tr>
<tr>
<td>onmode -C</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -d</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -D</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -e keyword</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -F</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -j</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -k</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -ku</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -l</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -m</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -M kilobytes</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -n</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -O</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -p { +</td>
<td>-</td>
</tr>
</tbody>
</table>
Table 11-1. onmode options (continued)

<table>
<thead>
<tr>
<th>onmode option</th>
<th>Administrative API command string</th>
</tr>
</thead>
<tbody>
<tr>
<td>onmode -Q</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -r</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -R</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -s</td>
<td>ONMODE, QUIESCENT</td>
</tr>
<tr>
<td>onmode -S</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -u</td>
<td>ONMODE, QUIESCENT IMMEDIATE</td>
</tr>
<tr>
<td>onmode -W</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -wf</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -wm</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -Y {0</td>
<td>1}</td>
</tr>
<tr>
<td>onmode -z session_id</td>
<td>ONMODE</td>
</tr>
<tr>
<td>onmode -Z address</td>
<td>ONMODE</td>
</tr>
</tbody>
</table>

onmode Syntax


Notes:
1. See the IBM Informix Migration Guide for onstat —b description.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-y</td>
<td>Causes the database server to automatically respond yes to all prompts</td>
<td>None.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>--------------------</td>
</tr>
<tr>
<td>-V</td>
<td>Displays the software version number and the serial number</td>
<td>See “Obtaining Utility Version Information” on page 6-2.</td>
</tr>
<tr>
<td>-version</td>
<td>Displays the build version, host, OS, number and date, as well as the GLS version</td>
<td>See “Obtaining Utility Version Information” on page 6-2.</td>
</tr>
</tbody>
</table>

**onmode -a: Add a shared-memory segment**

**Syntax:**

```
>>> onmode -a seg_size
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a seg_size</td>
<td>Allows you to add a new virtual shared-memory segment. Size is specified in kilobytes</td>
<td><strong>Restrictions:</strong> The value of seg_size must be a positive integer. It must not exceed the operating-system limit on the size of shared-memory segments.</td>
</tr>
</tbody>
</table>

Ordinarily, you do not need to add segments to the virtual portion of shared memory because the database server automatically adds segments as they are needed. However, as segments are added, the database server might reach the operating-system limit for the maximum number of segments before it acquires the memory that it needs. This situation typically occurs when SHMADD is set so small that the database server exhausts the number of available segments before it acquires the memory that it needs for some operation.

If you manually add a segment that is larger than the segment specified by SHMADD, you can avoid exhausting the operating-system limit for segments but still meet the need that the database server has for additional memory.

Use ADD MEMORY as the Administrative API command string for onmode -a. Use ONMODE as the Administrative API command string for onmode -a seg_size.

**onmode -BC: Allow large chunk mode**

**Syntax:**

```
>>> onmode -BC 1
>>> onmode -BC 2
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-BC 1</td>
<td>Enables support of large chunks, large offsets that are greater than 2 GB, and allows up to 32,768 chunks per dbspace.</td>
<td>This option allows large chunks to be created. Reversion without dropping the dbspace is possible if no chunks are larger than 2 GB. Dbspaces and blobspaces without chunks greater than 2 GB remain in the old format. After a chunk larger than 2 GB is added to a dbspace or blobspace then all chunks added or altered in that dbspace or blobspace are in the new format. See your IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>--------------------</td>
</tr>
</tbody>
</table>
| -BC 2   | Allows large-chunk-only mode for all dbspaces. | Reversion is not possible. Enables the 9.4 large chunk feature for all dbspaces and blobspaces. Any chunk or offset added or modified has the new format. Existing chunks that you do not alter remain in the old format.  
See your IBM Informix Administrator’s Guide |

The **onmode -BC** (backward-compatible) commands are useful if you have converted from Dynamic Server 9.40 (small chunk mode) to Dynamic Server 10.0 or later. When Dynamic Server 10.0 or later is first initialized (with the **oninit -iyv** command), by default it comes online with large chunk mode already fully enabled. Reversion is not possible. In the case of a newly initialized instance of Dynamic Server 10.0 or later, the **onmode -BC** commands will return an error.

Use ONMODE as the Administrative API **command** string for **onmode -BC**.

**Note:** After executing the **onmode -BC** command, perform a complete system level-0 backup.

**onmode -c: Force a checkpoint**

**Syntax:**

```
ONMODE -c <block|unblock>
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-c</td>
<td>Forces a checkpoint that flushes the buffers to disk</td>
<td>You can use the -c option to force a sync checkpoint if the most recent checkpoint record in the logical log was preventing the logical-log file from being freed (status U-B-L).</td>
</tr>
</tbody>
</table>
| block   | Blocks the database server from any transactions | While the database server is blocked, users can access it in read-only mode. Use this option to perform an external backup on Dynamic Server.  
For more information, see the *IBM Informix Backup and Restore Guide*. |
| unblock | Unblocks the database server | When the database server is unblocked, data transactions and normal database server operations can resume. Use this option after you complete an external backup on Dynamic Server.  
For more information, see the *IBM Informix Backup and Restore Guide*. |

Use **CHECKPOINT** as the Administrative API **command** command for **onmode -c**. Use ONMODE as the Administrative API **command** string for **onmode -c block** or **onmode -c unblock**.
### onmode -C: Control the B-tree scanner

**Syntax:**

```plaintext
>>> onmode -C (yielding syntax)
    start
    count
    stop count
    kill count
    threshold size
    duration num
    rangesize size
    alice num
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-C</code></td>
<td>Controls the B-tree scanner for cleaning indexes of deleted items</td>
<td>There is no limit to the number of threads that can run at one time. However, there is a limit of 128 threads that can be started at one time. If, for example, you wanted 150 threads to run, you could execute two commands: <code>onmode -C 100</code> and <code>onmode -C 50</code>.</td>
</tr>
<tr>
<td><strong>start</strong></td>
<td>Starts additional B-tree scanner threads.</td>
<td>If <code>count</code> is not specified, a <code>count</code> of 1 is assumed. There is no limit on the number of scanner threads that can be specified.</td>
</tr>
<tr>
<td><strong>stop</strong></td>
<td>Stops B-tree scanner threads.</td>
<td>If <code>count</code> is not specified, a <code>count</code> of 1 is assumed. Stopping all index scanners prevents all index cleaning.</td>
</tr>
<tr>
<td><strong>kill</strong></td>
<td>Stops B-tree scanner threads.</td>
<td>Either of these commands stop the B-tree scanner.</td>
</tr>
<tr>
<td><strong>threshold</strong></td>
<td>Sets the minimum number of deleted items an index must encounter before an index is placed on the hot list.</td>
<td>Once all indexes above the threshold have been cleaned and there is no other work for the B-tree scanner to do, the indexes below the threshold are added to the hot list.</td>
</tr>
<tr>
<td><strong>duration</strong></td>
<td>The number of seconds that the hot list is valid.</td>
<td>After this number of seconds expires, the hot list will be rebuilt by the next available B-tree scanner thread, even if unprocessed items are on the list. Scanners currently processing requests are not interrupted.</td>
</tr>
<tr>
<td><strong>rangesize</strong></td>
<td>Determines the size of an index before index range cleaning is enabled.</td>
<td>A size of -1 can be used to disable range scanning.</td>
</tr>
<tr>
<td><strong>alice</strong></td>
<td>Sets the system's <code>alice</code> mode.</td>
<td>Valid <code>num</code> values range from 0 (OFF) to 12.</td>
</tr>
</tbody>
</table>

The B-tree scanner has statistical information which tracks index efficiency and how much extra work the index currently places on the server. Based on the amount of extra work the index has accomplished because of committed deleted index items, the B-tree scanner develops an ordered list of indexes which have caused the server to do extra work, called the hot list. The index causing the highest amount of extra work is cleaned first and the rest of the indexes are cleaned in descending order. The DBA can allocate cleaning threads dynamically, thus allowing for configurable workloads.

Use ONMODE as the Administrative API `command` string for `onmode -C`. 
### onmode -d: Set data-replication types

**Syntax:**

```
onmode -d standard
    primary dbservername
    secondary
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d</code></td>
<td>Used to set the High-Availability Data-Replication type, either standard, primary, or secondary, as the following sections describe</td>
<td>You can use the <code>-d primary</code> and <code>-d secondary</code> options only when the database server is in quiescent mode. You can use the <code>-d standard</code> option when the database server is in quiescent, online, or read-only mode.</td>
</tr>
<tr>
<td><code>dbservername</code></td>
<td>Identifies the database server name of the primary or secondary database server</td>
<td>The <code>dbservername</code> argument must correspond to the DBSERVERNAME parameter in the ONCONFIG file of the intended secondary database server. It should <em>not</em> correspond to one of the database servers that the DBSERVERALIASES parameter specifies. The <code>dbservername</code> argument of the other database server in the data-replication pair and the type of a database server (standard, primary, or secondary) is preserved after reinitialization of shared memory. For more information, see range of values for the DBSERVERNAME configuration parameter in &quot;DBSERVERNAME&quot; on page 1-25</td>
</tr>
</tbody>
</table>

**Using the `-d standard` Option**

The `-d standard` option drops the connection between database servers in a data replication pair (if one exists) and sets the database server type of the current database server to standard. This option does not change the mode or type of the other database server in the pair.

**Using the `-d primary dbservername` Option**

The `-d primary dbservername` option sets the database server type to primary and attempts to connect with the database server that `dbservername` specifies. If the connection is successful, data replication is turned on. The primary database server goes into online mode, and the secondary database server goes into read-only mode. If the connection is not successful, the database server comes to on-line mode, but data replication is not turned on.

**Using the `-d secondary dbservername` Option**

The `-d secondary dbservername` option sets the database server type to secondary and attempts to connect with the database server that `dbservername` specifies. If the connection is successful, data replication is turned on. The primary database server goes online, and the secondary database server goes into read-only mode. If the connection is not successful, the database server comes to read-only mode, but data replication is not turned on.

Use ONMODE as the Administrative API *command* string for onmode -d.
For other `onstat -d` information, see "onmode -d: Set secondary server characteristics" and "onmode -d: Replicate an index with data-replication" on page 11-9.

### onmode -d: Set secondary server characteristics

**Syntax:**

```
$onmode -d
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-d</code></td>
<td>Used to create, modify, or delete secondary servers in high-availability configurations</td>
<td>The <code>servername</code> argument can be <code>DBSERVERNAME</code>, <code>DBSERVERALIASES</code> and ER group name. For more information, see range of values for <code>DBSERVERNAME</code> configuration parameter in &quot;DBSERVERNAME&quot; on page 1-27 and <code>DBSERVERALIASES</code> in 1-26.</td>
</tr>
<tr>
<td><code>servername</code></td>
<td>Identifies the database server name of the secondary database server</td>
<td></td>
</tr>
<tr>
<td><code>add RSS</code></td>
<td>Adds an RS secondary server</td>
<td></td>
</tr>
<tr>
<td><code>change RSS</code></td>
<td>Change an RS secondary server password</td>
<td>The password can only be changed before the RS secondary server is connected to the primary server. Once the RS secondary server connects to the primary server, the password cannot be changed.</td>
</tr>
<tr>
<td><code>delete RSS</code></td>
<td>Removes an RS secondary server definition</td>
<td></td>
</tr>
<tr>
<td><code>set SDS primary</code></td>
<td>Defines the alias server as a shared disk primary server</td>
<td></td>
</tr>
<tr>
<td><code>clear SDS primary</code></td>
<td>Enables the shared disk environment. The server name specified by alias no longer acts as an SD primary server</td>
<td></td>
</tr>
<tr>
<td><code>password</code></td>
<td>Specifies the secondary server password</td>
<td>The password is used only during the first connection attempt. After the primary and secondary server have connected, the password cannot be changed.</td>
</tr>
<tr>
<td><code>source_servername</code></td>
<td>Specifies the name of the primary server in a high-availability configuration</td>
<td></td>
</tr>
<tr>
<td><code>make primary</code></td>
<td>Creates a primary server</td>
<td></td>
</tr>
</tbody>
</table>
If the `force` option is specified, then the operation is performed without requiring that the secondary server is connected to the current primary server. If the `force` option is not specified, then the operation must be coordinated with the current primary server. The `force` option should only be used when the DBA is certain that the current primary server is not active; otherwise, the shared disk subsystem can become corrupted.

Use `ONMODE` as the Administrative API command string for `onmode -d`.

For other `onstat -d` information, see "onmode -d: Set data-replication types" on page 11-7 and "onmode -d: Replicate an index with data-replication."

### onmode -d: Replicate an index with data-replication

#### Syntax:

```
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>Specifies how indexes are replicated to a High-Availability Data-Replication (HDR) secondary server when an index on the secondary server becomes corrupt</td>
<td>You can use the <code>onmode -d idxauto</code> and <code>-d index</code> commands while the server is in online mode.</td>
</tr>
<tr>
<td>idxauto</td>
<td>Enables automatic index replication when an index on a secondary server becomes corrupt</td>
<td>Use <code>onmode -d idxauto</code> to overwrite the value of the DRIDXAUTO configuration parameter within a session. For more information on DRIDXAUTO, see &quot;DRIDXAUTO&quot; on page 1-35. For more information on replicating indexes, see the chapter on using HDR in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>index</td>
<td>Replicates an index from a primary to a secondary server</td>
<td>If you detect a corrupt index on a secondary server, use the <code>onmode -d index</code> command to start replication of the index from the primary to the secondary server.</td>
</tr>
<tr>
<td>database</td>
<td>Specifies the database containing the index to replicate</td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>index</td>
<td>Specifies the name of the index to replicate</td>
<td>Index must exist on table and in database specified. Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>owner</td>
<td>Specifies the owner of a table</td>
<td>You must specify the current owner of the table. Syntax must conform to the Table Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>table</td>
<td>Specifies the name of the table on which the index is based</td>
<td>Syntax must conform to the Table Name segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
</tbody>
</table>
The -d idxauto and the -d index options provide methods to replicate an index to a secondary server containing a corrupted index. The base table will be locked during the transfer of an index. The alternative to using these options is to drop and rebuild the corrupt index on the primary server.

In the case of a fragmented index with one corrupt fragment, the -d idxauto option only transfers the single affected fragment, whereas the -d index option transfers the whole index.

Use ONMODE as the Administrative API command string for onmode -d.

For other onstat -d information, see "onmode -d: Set secondary server characteristics" on page 11-8 and "onmode -d: Set data-replication types" on page 11-7.

### onmode -D, -M, -Q, -S: Change decision-support parameters

#### Syntax:

```
  onmode -D max_priority
  -M kilobytes
  -Q queries
  -S scans
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-D max_priority</td>
<td>Changes the value of MAX_PDQPRIORITY</td>
<td>This value must be an unsigned integer between 0 and 100. Specify max_priority as a factor to temper user requests for PDQ resources. For information on parameters used for controlling PDQ, see &quot;MAX_PDQPRIORITY&quot; on page 1-62 and the IBM Informix Performance Guide.</td>
</tr>
<tr>
<td>-M kilobytes</td>
<td>Changes the value of DS_TOTAL_MEMORY</td>
<td>This value has a platform-dependent upper limit. The value for 32-bit systems must be an unsigned integer between 128 * DS_MAX QUERIES and 1,048,576. On 64-bit systems, the limit is generally higher and varies with the operating system. On HP 9000 platforms, for example, the maximum value is 4,294,967,296. Specify kilobytes for the maximum amount of memory available for parallel queries. For more information, see &quot;DS_TOTAL_MEMORY&quot; on page 1-39 and the IBM Informix Performance Guide.</td>
</tr>
<tr>
<td>-Q queries</td>
<td>Changes the value of DS_MAX QUERIES</td>
<td>This value must be an unsigned integer between 1 and 8,388,608. Specify queries for the maximum number of concurrently executing parallel queries. For information on parameters used for controlling PDQ, see &quot;DS_MAX QUERIES&quot; on page 1-37 and the IBM Informix Performance Guide.</td>
</tr>
</tbody>
</table>
These options allow you to change configuration parameters while the database server is online. The new values affect only the current instance of the database server; the values are not recorded in the ONCONFIG file. If you shut down and restart the database server, the values of the parameters revert to the values in the ONCONFIG file. For more information about these configuration parameters, see Chapter 1, “Configuration Parameters,” on page 1-1.

To check the current values for the MAX_PDQPRIORITY, DS_TOTAL_MEMORY, DS_MAX_SCANS, DS_MAX_QUERIES, and the DS_NONPDQ_QUERY_MEM configuration parameters, use onstat -g mgm. See “onstat -g mgm: Print MGM resource information” on page 15-61.

Use ONMODE as the Administrative API command string for onmode -D, onmode -M, onmode -Q, or onmode -S.

### onmode -e: Change usage of the SQL statement cache

**Syntax:**

```
onmode -e mode```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>onmode -e ENABLE</td>
<td>Enables the SQL statement cache</td>
<td>User sessions use the cache only when they perform either of the following actions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Set the environment variable STMT_CACHE to 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Execute the SQL statement SET STATEMENT CACHE ON</td>
</tr>
<tr>
<td>onmode -e FLUSH</td>
<td>Flushes the statements that are not in use from the SQL</td>
<td>The onstat -g ssc ref_cnt field shows 0.</td>
</tr>
<tr>
<td>onmode -e OFF</td>
<td>Turns off the SQL statement cache</td>
<td>No statements are cached.</td>
</tr>
<tr>
<td>onmode -e ON</td>
<td>Turns on the SQL statement cache</td>
<td>All statements are cached unless the user turns it off with one of the following actions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Set the environment variable STMT_CACHE to 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Execute the SQL statement SET STATEMENT CACHE OFF</td>
</tr>
</tbody>
</table>
The onmode -e changes are in effect for the current database server session only. When you restart the database server, it uses the default STMT_CACHE parameter value in the ONCONFIG file.

Use ONMODE as the Administrative API command string for onmode -e.

---

**onmode -F: Free unused memory segments**

**Syntax:**

```
> onmode -F
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-F</td>
<td>Frees unused memory segments</td>
<td>None.</td>
</tr>
</tbody>
</table>

When you execute onmode -F, the memory manager examines each memory pool for unused memory. When the memory manager locates blocks of unused memory, it immediately frees the memory. After the memory manager checks each memory pool, it begins checking memory segments and frees any that the database server no longer needs.

It is recommended that you run onmode -F from an operating-system scheduling facility regularly and after the database server performs any function that creates additional memory segments, including large index builds, sorts, or backups.

Running onmode -F causes a significant degradation of performance for any users that are active when you execute the utility. Although the execution time is brief (1 to 2 seconds), degradation for a single-user database server can reach 100 percent. Systems with multiple CPU virtual processors experience proportionately less degradation.

To confirm that onmode freed unused memory, check your message log. If the memory manager frees one or more segments, it displays a message that indicates how many segments and bytes of memory were freed.

Use ONMODE as the Administrative API command string for onmode -F.

---

**onmode -k, -m, -s, -u, -j: Change database server mode**

**Syntax:**

```
> onmode -k
> onmode -m
> onmode -s
> onmode -u
> onmode -j
```

---

**IBM Informix Dynamic Server Administrator’s Reference**
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
</table>
| -k      | Takes the database server to offline mode and removes shared memory | To reinitialize shared memory, shut down and restart the database server.

See "Taking the Database Server to Offline Mode with the -k Option" on page 11-13 |
| -m      | Takes the database server from quiescent or administration mode to online mode | See "Bringing the Database Server Online with the -m Option" on page 11-14 |
| -s      | Shuts down the database server gracefully | Users who are using the database server are allowed to finish before the database server comes to quiescent mode, but no new connections are allowed. When all processing is finished, -s takes the database server to quiescent mode. The -s option leaves shared memory intact.

See "Shutting Down the Database Server Gracefully with the -s Option" on page 11-14 |
| -u      | Shuts down the database server immediately | This option brings the database server to quiescent mode without waiting for users to finish their sessions. Their current transactions are rolled back, and their sessions are terminated.

See "Shutting Down the Database Server Immediately with the -u Option" on page 11-14 |
| -j      | Puts the database server into administration mode | This option brings the database server to administration mode, allowing the informix user all functions including the issuance of SQL and DDL commands. The -j -U option enables the DBSA to designate specific users (in addition to the informix user) to access the database server.

See your IBM Informix Administrator’s Guide. |

The following sections describe the options that take the database server from one mode to another.

**Taking the Database Server to Offline Mode with the -k Option**

The **onmode -k** option takes the database server to offline mode and removes database server shared memory. Use SHUTDOWN or ONMODE as the Administrative API command string for **onmode -k**.

A prompt asks for confirmation. Another prompt asks for confirmation to kill user threads before the database server comes offline. If you want to eliminate these prompts, execute the -y option with the -s option.

This option does not kill all client sessions. Use the -u option to avoid hanging client sessions or virtual server processes.

**Important:** When you use the **onmode -k** command to shut down the database server, utilities that are waiting for a user response might not terminate. For example, **ontape** might be waiting for another tape, **onstat -i** might be waiting for a user response, or **onspaces** might be waiting for **y** or **n** to continue. If this problem occurs, use **onmode -uk** or -uky instead to roll back work before removing shared memory. For more information, see the descriptions of other options on this page.
Bringing the Database Server Online with the -m Option

The -m option brings the database server online from quiescent mode.

Shutting Down the Database Server Gracefully with the -s Option

The -s option causes a graceful shutdown. Users who are using the database server are allowed to finish before the database server comes to quiescent mode, but no new connections are allowed. When all processing is finished, -s takes the database server to quiescent mode. The -s option leaves shared memory intact.

A prompt asks for confirmation. If you want to eliminate this prompt, execute the -y option with the -s option.

Use QUIESCENT as the Administrative API command string for onmode -s.

Shutting Down the Database Server Immediately with the -u Option

The -u option causes immediate shutdown. This option brings the database server to quiescent mode without waiting for users to finish their sessions. Their current transactions are rolled back, and their sessions are terminated.

A prompt asks for confirmation. Another prompt asks for confirmation to kill user threads before the database server comes to quiescent mode. If you want to eliminate these prompts, execute the -y option with the -s option.

Changing the Database Server to Administration Mode with the -j Option

The -j option puts the database server into the administration mode and allows only the DBSA group and the user informix to connect to the server. The -j option allows a DBSA to have the server in a fully functional mode to perform maintenance.

The -j -U option enables the DBSA to grant individual users access to the database server in administration mode. Once connected, these individual users can execute any SQL or DDL command. When the server is changed to administration mode, all sessions for users other than user informix, the DBSA group users, and those identified in the onmode -j -U command lose their database server connection.

The following example enables three individual users to connect to the database server and have database server access until the database server mode changes to offline, quiescent or online mode:

```
onmode -j -U karin,sarah,andy
```

Access for individual users can also be removed by executing onmode -j -U and removing their name from the new list of names in the command. For example, in the following commands, the first command grants only Karin access, the second command grants Karin and Sarah access, and the third command grants only Sarah access (and removes access from Karin).

```
onmode -j -U karin
onmode -j -U karin,sarah
onmode -j -U sarah
```
To allow user informix and the DBSA group user to retain their database server access in administration mode and remove all single users from accessing the database server, use the following command:

`onmode -j -u`  

For information on designating single users in administration mode using a configuration parameter, see "ADMIN_MODE_USERS" on page 1-13.

### Changing Database Server Mode with ON-Monitor (UNIX)

You can also use ON-Monitor options to change the database server mode. The following table shows ON-Monitor options that are equivalent to the `onmode` options.

<table>
<thead>
<tr>
<th>onmode Option</th>
<th>ON-Monitor Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>-k</td>
<td>Take-Offline</td>
</tr>
<tr>
<td>-m</td>
<td>On-Line</td>
</tr>
<tr>
<td>-s</td>
<td>Graceful-Shutdown</td>
</tr>
<tr>
<td>-u</td>
<td>Immediate-Shutdown</td>
</tr>
<tr>
<td>-j</td>
<td>Administration Mode</td>
</tr>
</tbody>
</table>

Use ONMODE as the Administrative API `command` string for `onmode -j`, `onmode -k`, `onmode -m`, `onmode -s`, and `onmode -u`. In addition, use SHUTDOWN, QUIESCENT and QUIESCENT IMMEDIATE as the Administrative API `command` strings for `onmode -k`, `onmode -s`, and `onmode -u`, respectively.

### onmode -l: Switch the logical-log file

**Syntax:**

```
>>> onmode -l
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
</table>
| -l      | Switches the current logical-log file to the next logical-log file | You must use `onmode` to switch to the next logical-log file.  
For information on switching to the next logical-log file, see the chapter on managing logical-log files in the IBM Informix Administrator's Guide. |

Use ONMODE as the Administrative API `command` string for `onmode -l`.

### onmode -n, -r: Change shared-memory residency

**Syntax:**

```
>>> onmode -n
```

```
>>> onmode -r
```

Chapter 11. The onmode Utility  11-15
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n</td>
<td>Ends forced residency of the resident portion of shared memory</td>
<td>This command does not affect the value of RESIDENT, the forced-residency parameter in the ONCONFIG file.</td>
</tr>
<tr>
<td>-r</td>
<td>Starts forced residency of the resident portion of shared memory</td>
<td>This command does not affect the value of RESIDENT, the forced-memory parameter in the ONCONFIG file.</td>
</tr>
</tbody>
</table>

**Important:** Set the RESIDENT parameter to 1 before you use the `onmode -r` or `-n` options.

For information on using the forced-residency parameter to turn residency on or off for the next time that you restart the database server, see the chapter on managing shared memory in the *IBM Informix Administrator’s Guide*.

Use ONMODE as the Administrative API command string for `onmode -n` or `onmode -r`.

**onmode -O: Override ONDBSPACEDOWN WAIT mode**

**Syntax:**

```
>>>onmode -O
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-O</td>
<td>Overrides the WAIT mode of the ONDBSPACEDOWN configuration parameter</td>
<td>None.</td>
</tr>
</tbody>
</table>

Use the `onmode -O` option only in the following circumstances:

- ONDBSPACEDOWN is set to WAIT.
- A disabling I/O error occurs that causes the database server to block all updating threads.
- You cannot or do not want to correct the problem that caused the disabling I/O error.
- You want the database server to mark the disabled dbspace as down and continue processing.

When you execute this option, the database server marks the dbspace responsible for the disabling I/O error as down, completes a checkpoint, and releases blocked threads. Then `onmode` prompts you with the following message:

This will render any dbspaces which have incurred disabling I/O errors unusable and require them to be restored from an archive.

Do you wish to continue?(y/n)

If `onmode` does not find any disabling I/O errors on noncritical dbspaces when you run the `-O` option, it notifies you with the following message:

There have been no disabling I/O errors on any noncritical dbspaces.

Use ONMODE as the Administrative API command string for `onmode -O`. 
### onmode -p: Add or remove virtual processors

**Syntax:**

```
--onmode -p number --
```

**Element** | **Purpose** | **Key Considerations** |
---|---|---|
-p number | Adds or removes virtual processors. The `number` argument indicates the number of virtual processors to add or remove. If this value is a negative integer, processors are removed. If this value is a positive integer, processors are added. | You can use the -p option only when the database server is in online mode, and you can add to only one class of virtual processors at a time. For more details, see "Adding and Dropping Virtual Processors" on page 11-18. If you are removing virtual processors, the maximum cannot exceed the actual number of processors of the specified type. If you are adding virtual processors, the maximum number depends on the operating system. For more information, see the chapter on using virtual processors in the `IBM Informix Administrator’s Guide`. |
AIO | Performs nonlogging disk I/O to cooked disk spaces | Also performs nonlogging I/O to raw disk spaces if kernel asynchronous I/O (KAIO) is not used. |
CPU | Runs all session threads and some system threads | It is recommended that the number of CPU VPs not be greater than the number of physical processors. If KAIO is used, performs I/O to raw disk spaces, including I/O to physical and logical logs. Runs thread for KAIO where available or a single poll thread. The database server uses the number of CPU VPs to allocate resources for parallel database queries (PDQ). If you drop CPU VPs, your queries will run significantly slower. The `Reinit` field of the `onstat -g mgm` output displays information on the number of queries that are waiting for running queries to complete after an `onmode -p` command. Also see the `IBM Informix Performance Guide`. |
ENCRIPT | Executes column-level encryption and decryption routines | Specify more ENCRYPT virtual processors if you have multiple encrypted columns. |
JVP | Executes Java user-defined routines in the Java Virtual Machine (JVM) | Specify more JVPs if you are running many Java UDRs. |
LIO | Writes to the logical-log files if they are in cooked disk space | Use two LIO virtual processors only if the logical logs are in mirrored dbspaces. The database server allows a maximum of two LIO virtual processors. |
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIO</td>
<td>Writes to the physical log if it is in cooked disk space</td>
<td>Use two PIO virtual processors only if the physical log is in a mirrored dbspace. The database server allows a maximum of two PIO virtual processors.</td>
</tr>
<tr>
<td>SHM</td>
<td>Performs shared-memory communication</td>
<td>You can use the SHM virtual processor even if the database server is not configured for shared-memory communication.</td>
</tr>
<tr>
<td>SOC</td>
<td>Uses sockets to perform network communications</td>
<td>You can use the SOC virtual processor only if the database server is configured for network connections through sockets.</td>
</tr>
<tr>
<td>STR</td>
<td>Performs stream pipe connections</td>
<td></td>
</tr>
<tr>
<td>TLI</td>
<td>Uses the Transport Layer Interface (TLI) to perform network communication</td>
<td>You can use the TLI virtual processor only if the database server is configured for network connections through TLI.</td>
</tr>
</tbody>
</table>
| vpclass | Names a user-defined virtual processor class | Use the VPCLASS parameter in the ONCONFIG to define the user-defined virtual-processor class. Specify more user-defined virtual processors if you are running many UDRs.  
On Windows, you can have only one user-defined virtual processor class at a time. Omit the number parameter in the onmode -p vpclass command.  
For more information on extension classes, see "VPCLASS" on page 1-103. |

### Adding and Dropping Virtual Processors

The following rules about adding or dropping virtual processors apply:

- You can add but not drop virtual processors of the AIO, PIO, LIO, TLI, SHM, SOC, and STR classes.
- You cannot add or drop virtual processors of the OPT, ADM, ADT, and MSC classes. The database server adds them automatically.
- You can add or drop virtual processors of the CPU, ENCRYPT, JVP, and user-defined (vpclass) classes.

#### Windows Only

- On Windows, you can add a virtual processor of any class, but you cannot drop virtual processors.

--- End of Windows Only ---

### Dropping Virtual Processors Automatically

Table 11-2 shows the virtual processors that the database server starts automatically. You cannot add or drop these virtual processors with the onmode -p command. To drop these virtual processors, shut down and restart the database server.

**Table 11-2. Virtual-Processor Classes That the Database Server Starts Automatically**

<table>
<thead>
<tr>
<th>Virtual-Processor Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADM</td>
<td>Performs administrative functions</td>
</tr>
<tr>
<td>ADT</td>
<td>Runs auditing processes The database server starts one virtual processor in the audit class when you turn on audit mode by setting the ADTMODE parameter in the ONCONFIG file.</td>
</tr>
</tbody>
</table>
Virtual-Processor Classes That the Database Server Starts Automatically (continued)

<table>
<thead>
<tr>
<th>Virtual-Processor Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC</td>
<td>Services requests for system calls that require a large stack. The database server starts this virtual processor automatically.</td>
</tr>
<tr>
<td>OPT</td>
<td>Performs I/O to the optical disk. The database server starts one OPT virtual processor when you use the Optical Subsystem.</td>
</tr>
</tbody>
</table>

Monitoring Poll Threads with onstat

While the database server is online, you cannot drop a CPU virtual processor that is running a poll thread. To identify poll threads that run on CPU virtual processors, use the following command:

```bash
onstat -g ath | grep 'cpu.*poll'
```

The following `onstat -g ath` output shows two CPU virtual processors with poll threads. In this situation, you cannot drop to fewer than two CPU virtual processors.

```
tid tcb rstcb prty status vp-class name
8 a362b90 0 2 running 1cpu tlitcppoll
9 a36e8e0 0 2 cond wait arrived 3cpu
```

For more information on the types of virtual processors, see the chapter on virtual processors and threads in the *IBM Informix Administrator’s Guide*.

Use ONMODE as the Administrative API command string for `onmode -p`.

**onmode -R: Regenerate .infos File**

**Syntax:**

```
onmode -R
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-R</td>
<td>Re-creates the <code>.infos.dbservername</code> file</td>
<td>Before you use the -R option, set the <code>INFORMIXSERVER</code> environment variable to match the <code>DBSERVERNAME</code> parameter from the <code>ONCONFIG</code> file. Do not use the -R option if <code>INFORMIXSERVER</code> is one of the <code>DBSERVERALIAS</code> names. For more information, see <a href="#">“.infos.dbservername” on page A-6</a>.</td>
</tr>
</tbody>
</table>

The database server uses information from the `.infos.dbservername` file when it accesses utilities. The database server creates and manages this file, and you should never need to do anything to the file. However, if `.infos.dbservername` is accidentally deleted, you must either recreate the file or shut down and restart the database server.

Use ONMODE as the Administrative API command string for `onmode -R`. 

---

Chapter 11. The onmode Utility 11-19
onmode -W: Change settings for the SQL statement cache

Syntax:

```bash
  onmode -W [STMT_CACHE_HITS=hits] [STMT_CACHE_NOLIMIT=value]
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>STMT_CACHE_HITS hits</td>
<td>Specifies the number of hits (references) to a statement before it is fully inserted in the SQL statement cache. Set hits to 1 or more to exclude ad hoc queries from entering the cache.</td>
<td>You can only increase or reset the value of STMT_CACHE_HITS. The new value displays in the #hits field of the onstat-g ssc output. If hits = 0, the database server inserts all qualified statements and its memory structures in the cache. If hits &gt; 0 and the number of times the SQL statement has been executed is less than STMT_CACHE_HITS, the database server inserts key-only entries in the cache. It inserts qualified statements in the cache after the specified number of hits have been made to the statement. <strong>ONCONFIG</strong> Parameter: STMT_CACHE_HITS</td>
</tr>
<tr>
<td>STMT_CACHE_NOLIMIT value</td>
<td>Controls whether statements are inserted in the SQL statement cache.</td>
<td>If value = 0, the database server inserts statements in the cache. If value = 1, the database server always inserts statements in the cache. If none of the queries are shared, turn off STMT_CACHE_NOLIMIT to prevent the database server from allocating a large amount of memory for the cache. <strong>ONCONFIG</strong> Parameter: STMT_CACHE_NOLIMIT</td>
</tr>
</tbody>
</table>

**SQL Statement Cache Examples**

The following are examples of onmode -W commands for changing SQL statement cache (SSC) settings. The changes are in effect for the current database server session only and do not change the ONCONFIG values. When you restart the database server, it uses the default SSC settings, if not specified in the ONCONFIG file, or the ONCONFIG settings. To make the changes permanent, set the appropriate configuration parameter.

```bash
onmode -W STMT_CACHE_HITS 2  # number of hits before statement is inserted into SSC
onmode -W STMT_CACHE_NOLIMIT 1  # always insert statements into the cache
```

Use ONMODE as the Administrative API command string for onmode -W.

onmode -wf, -wm: Dynamically change certain configuration parameters

Syntax:

```bash
  onmode -wf config_param=value
  onmode -wm config_param=value
```
Use **onmode -wf** or **onmode -wm** to dynamically change certain configuration parameters.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-wf</td>
<td>Updates the value of the specified configuration parameter in the ONCONFIG file.</td>
<td>None.</td>
</tr>
<tr>
<td>-wm</td>
<td>Dynamically sets the value of the specified configuration parameter for the current session.</td>
<td>None.</td>
</tr>
</tbody>
</table>

`config_param=value`  
The configuration parameter and its new value. The following configuration parameters can be specified:  
- `ADMIN_MODE_USERS`  
- `AUTO_AIOVPS`  
- `AUTO_CKPTS`  
- `DS_MAX_QUERIES`  
- `DS_MAX_SCANS`  
- `DS_NONPDQ_QUERY_MEM`  
- `DS_TOTAL_MEMORY`  
- `EXPLAIN_STAT`  
- `LISTEN_TIMEOUT`  
- `LOG_INDEX_BUILDS`  
- `MAX_INCOMPLETE_CONNECTIONS`  
- `MAX_PDQPRIORITY`  
- `ONLIDX_MAXMEM`  
- `RESIDENT`  
- `RTO_SERVER_RESTART`  
- `SDS_ENABLE`  
- `SDS_TIMEOUT`  
- `TEMPTAB_NOLOG`  
- `USELASTCOMMITTED`  
- `VP_MEMORY_CACHE_KB`  

See [Chapter 1, "Configuration Parameters," on page 1-1](#).

Use **ONMODE** as the Administrative API command string for **onmode -wf** or **onmode -wm**.

**onmode -wm: Change LRU tuning status**

**Syntax:**

```
  onmode -wm AUTO_LRU_TUNING -i [min, max]
```

Use **onmode -wm** to change LRU tuning status.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-wm</td>
<td>Dynamically sets the value of the specified configuration parameter for the current session.</td>
<td>None.</td>
</tr>
</tbody>
</table>
You can use the SET EXPLAIN statement to display the query plan of the optimizer, an estimate of the number of rows returned, and the relative cost of the query. When you use the `onmode -Y` command to turn on SET EXPLAIN, the output is displayed in the `sqexplain.out.sessionid` file. If an `sqexplain.out` file already exists, the database server reads that file until an administrator turns off the SET EXPLAIN for the session.

Use ONMODE as the Administrative API command string for `onmode -wm`.

### onmode -Y: Dynamically change SET EXPLAIN

**Syntax:**

```bash
onmode -Y sessionid
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-Y</td>
<td>Dynamically change the value of the SET EXPLAIN statement.</td>
<td>None.</td>
</tr>
<tr>
<td>sessionid</td>
<td>Identifies the specific session.</td>
<td>None.</td>
</tr>
</tbody>
</table>

The `onmode -Y` command dynamically changes the value of the SET EXPLAIN statement for an individual session. The following invocations are valid with this command:

<table>
<thead>
<tr>
<th>Invocation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>onmode -Y sessionid 2</code></td>
<td>Turns SET EXPLAIN on for <code>sessionid</code> and displays the query plan only</td>
</tr>
<tr>
<td><code>onmode -Y sessionid 1</code></td>
<td>Turns SET EXPLAIN on for <code>sessionid</code></td>
</tr>
<tr>
<td><code>onmode -Y sessionid 0</code></td>
<td>Turns SET EXPLAIN off for <code>sessionid</code></td>
</tr>
</tbody>
</table>

For more information on using the SET EXPLAIN statement, see the *IBM Informix Guide to SQL: Syntax*. For more information on interpreting the `sqexplain.out` file to improve query performance, see the *IBM Informix Performance Guide*.
Use ONMODE as the Administrative API command string for onmode -Y.

**onmode -z: Kill a database server session**

**Syntax:**

```
>> onmode -z sid
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-z sid</code></td>
<td>Kills the session that you specify in <code>sid</code></td>
<td>This value must be an unsigned integer greater than 0 and must be the session identification number of a currently running session.</td>
</tr>
</tbody>
</table>

To use the `-z` option, first obtain the session identification (`sessid`) with `onstat -u`, then execute `onmode -z`, substituting the session identification number for `sid`.

When you use `onmode -z`, the database server attempts to kill the specified session. If the database server is successful, it frees any resources that the session holds. If the database server cannot free the resources, it does not kill the session.

If the session does not exit the section or release the latch, the database server administrator can take the database server offline, as described in “Taking the Database Server to Offline Mode with the -k Option” on page 11-13 to close all sessions.

Use ONMODE as the Administrative API command string for onmode -z.

**onmode -Z: Kill a distributed transaction**

**Syntax:**

```
>> onmode -Z address
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-Z address</code></td>
<td>Kills a distributed transaction associated with the shared-memory address <code>address</code></td>
<td>This argument must be the address of an ongoing distributed transaction that has exceeded the amount of time that TXTIMEOUT specifies. The address must conform to the operating-system-specific rules for addressing shared-memory. (The address is available from <code>onstat -x</code> output.)</td>
</tr>
</tbody>
</table>

This option is not valid until the amount of time that the ONCONFIG parameter TXTIMEOUT specifies has been exceeded. The `-Z` option should rarely be used and only by an administrator of a database server involved in distributed transactions.

For information on initiating independent actions in a two-phase commit protocol, see the chapter on multiphase commit protocols in the *IBM Informix Administrator’s Guide*.

Distributed transactions provide the ability to query data on different database servers.
**Warning:** If applications are performing distributed transactions, killing one of the distributed transactions can leave your client/server database system in an inconsistent state. Try to avoid this situation.

Use ONMODE as the Administrative API *command* string for **onmode -Z**.
Chapter 12. The ON-Monitor Utility

Using ON-Monitor (UNIX) ................................................................. 12-1
Navigating ON-Monitor and Using Help ..................................... 12-1
Executing Shell Commands Within ON-Monitor ......................... 12-1
ON-Monitor Screen Options ......................................................... 12-2
Setting Configuration Parameters in ON-Monitor ......................... 12-3

Using ON-Monitor (UNIX)

Use the ON-Monitor utility to perform various administrative tasks. This section provides a quick reference for the ON-Monitor screens. To start ON-Monitor, execute the following command from the operating-system prompt:

```
onmonitor```

If you are logged in as user informix or user root, the main menu appears. All users other than informix and root have access only to the Status menu.

The ON-Monitor main menu displays the following menus:

- Status menu
- Parameters menu
- Dbspaces menu
- Mode menu
- Force-Ckpt menu
- Archive menu
- Logical-Logs menu
- Exit option

These menus are shown on the following pages (Table 12-1 on page 12-2 through Table 12-7 on page 12-3).

To obtain ON-Monitor version information, execute the -V or -version command from the operating-system prompt. For complete information on version information, see “Obtaining Utility Version Information” on page 6-2

Navigating ON-Monitor and Using Help

All menus and screens in ON-Monitor function in the same way. For menus, use the arrow keys or SPACEBAR to scroll to the option that you want to execute and press RETURN, or press the first capitalized letter of the option (usually the first letter). When you move from one option to the next by pressing SPACEBAR or an arrow key, the option explanation (line 2 of the menu) changes.

If you want general instructions for a specific screen, press CTRL-W. If you need help to determine what you should enter in a field on the screen, use the TAB key to highlight the field and press CTRL-F or F2.

Some of the menus display ellipses (…) on the far right or left side. The ellipses indicate that you can move in the direction of the dots, using the arrow keys or SPACEBAR, to view other options.
Executing Shell Commands Within ON-Monitor

To execute a shell command from within ON-Monitor, type an exclamation point (!) followed by the command. For example, to list the files in the current directory, type the following command:

```
!ls
```

ON-Monitor Screen Options

Table 12-1. Status Menu

<table>
<thead>
<tr>
<th>Menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profile</td>
<td>Displays database server performance statistics</td>
</tr>
<tr>
<td>Userthreads</td>
<td>Displays the status of active user threads</td>
</tr>
<tr>
<td>Spaces</td>
<td>Displays status information about database server storage spaces and chunks</td>
</tr>
<tr>
<td>Databases</td>
<td>Displays the name, owner, and logging mode of the 100 first databases</td>
</tr>
<tr>
<td>Logs</td>
<td>Displays status information about the physical-log buffer, the physical log, the logical-log buffer, and the logical-log files</td>
</tr>
<tr>
<td>Archive</td>
<td>Displays a list of all backup tapes and logical-log files that you require to restore data using ontape</td>
</tr>
<tr>
<td>data-Replication</td>
<td>Displays High-Availability Data-Replication (HDR) status and configuration</td>
</tr>
<tr>
<td>Output</td>
<td>Stores the output of other status information in a specified file</td>
</tr>
<tr>
<td>Configuration</td>
<td>Copies the current database server configuration to a file</td>
</tr>
</tbody>
</table>

Table 12-2. Parameters Menu

<table>
<thead>
<tr>
<th>Menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize</td>
<td>Initializes database server disk space or modifies disk-space parameters</td>
</tr>
<tr>
<td>Shared-Memory</td>
<td>Initializes database server shared memory or modifies shared-memory parameters</td>
</tr>
<tr>
<td>performance</td>
<td>Specifies the number of virtual processors for each VP class</td>
</tr>
<tr>
<td>data-Replication</td>
<td>Specifies the HDR parameters</td>
</tr>
<tr>
<td>diagnostics</td>
<td>Specifies values for the diagnostics parameters</td>
</tr>
<tr>
<td>pdQ</td>
<td>Changes parameters for parallel database queries</td>
</tr>
<tr>
<td>Add-Log</td>
<td>Adds a logical-log file to a dbspace</td>
</tr>
<tr>
<td>Drop-Log</td>
<td>Drops a logical-log file from a dbspace</td>
</tr>
<tr>
<td>Physical-Log</td>
<td>Changes the size or the location of the database server physical log</td>
</tr>
</tbody>
</table>

Table 12-3. Dbspaces Menu

<table>
<thead>
<tr>
<th>Menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>Creates a dbspace</td>
</tr>
<tr>
<td>BLOBSpace</td>
<td>Creates a blobspace</td>
</tr>
<tr>
<td>Mirror</td>
<td>Adds mirroring to an existing storage space or ends mirroring for a storage space</td>
</tr>
<tr>
<td>Drop</td>
<td>Drops a storage space from the database server configuration</td>
</tr>
</tbody>
</table>
Table 12-3. Dbspaces Menu (continued)

<table>
<thead>
<tr>
<th>Menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Info</td>
<td>Displays the identification number, location, and fullness of each chunk assigned to a storage space</td>
</tr>
<tr>
<td>Add_chunk</td>
<td>Adds a chunk to a storage space</td>
</tr>
<tr>
<td>datasKip</td>
<td>Changes the database parameter</td>
</tr>
<tr>
<td>Status</td>
<td>Changes the status of a chunk in a mirrored pair</td>
</tr>
</tbody>
</table>

Table 12-4. Mode Menu

<table>
<thead>
<tr>
<th>Menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Startup</td>
<td>Initializes shared memory and takes the database server to quiescent mode</td>
</tr>
<tr>
<td>On-Line</td>
<td>Takes the database server from quiescent to online mode</td>
</tr>
<tr>
<td>Graceful-Shutdown</td>
<td>Takes the database server from online to quiescent mode so users can complete work</td>
</tr>
<tr>
<td>Immediate-Shutdown</td>
<td>Takes the database server from online to quiescent mode in 10 seconds</td>
</tr>
<tr>
<td>Take-Offline</td>
<td>Detaches shared memory and immediately takes the database server to offline mode</td>
</tr>
<tr>
<td>Add-Proc</td>
<td>Adds virtual processors</td>
</tr>
<tr>
<td>Drop-Proc</td>
<td>Drops virtual processors</td>
</tr>
<tr>
<td>Decision-support</td>
<td>Sets decision-support parameters dynamically</td>
</tr>
<tr>
<td>Administration</td>
<td>Tells the server to change into administration mode</td>
</tr>
</tbody>
</table>

Table 12-5. Force-Ckpt Menu

<table>
<thead>
<tr>
<th>Menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Force-Ckpt</td>
<td>Displays the time of the most-recent checkpoint or forces the database server to execute a checkpoint</td>
</tr>
</tbody>
</table>

Table 12-6. Archive Menu

<table>
<thead>
<tr>
<th>Menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tape-Parameters</td>
<td>Modifies the ontape parameters for the backup tape device</td>
</tr>
</tbody>
</table>

Table 12-7. Logical Logs Menu

<table>
<thead>
<tr>
<th>Menu</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Databases</td>
<td>Modifies the logging status of a database</td>
</tr>
<tr>
<td>Tape-Parameters</td>
<td>Modifies the ontape parameters for the logical-log backup tape device</td>
</tr>
</tbody>
</table>

Setting Configuration Parameters in ON-Monitor

Figure 12-1 shows which ONCONFIG parameters correspond to the Initialization screen.
Figure 12-1 shows which ONCONFIG parameters correspond to the Shared-Memory screen.

**Figure 12-2** shows which ONCONFIG parameters correspond to the Shared-Memory screen.

**Figure 12-2** shows which ONCONFIG parameters correspond to the Performance Tuning screen.

**Figure 12-3** shows which ONCONFIG parameters correspond to the Performance Tuning screen.

**Note:** Although Dynamic Server can support a shared memory segment that is larger than 4 gigabytes, ON-Monitor does not support a shared memory segment that is larger than 4 gigabytes. Therefore, the ON-Monitor screen cannot hold values that are larger than 4 gigabytes.

**Figure 12-4** shows which ONCONFIG parameters correspond to the Performance Tuning screen.
Figure 12-4 shows which ONCONFIG parameters correspond to the Data Replication screen.

**DATA REPLICATION PARAMETERS**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>ONCONFIG Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interval</td>
<td>DRINTERVAL</td>
</tr>
<tr>
<td>Timeout</td>
<td>DRTIMEOUT</td>
</tr>
<tr>
<td>Lost &amp; Found</td>
<td>DRLOSTFOUND</td>
</tr>
</tbody>
</table>

*Figure 12-4. Data-Replication Screen with Parameter Names*

Figure 12-5 shows which ONCONFIG parameters correspond to the Diagnostics screen.

**DIAGNOSTIC PARAMETERS**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>ONCONFIG Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Log</td>
<td>MSGPATH</td>
</tr>
<tr>
<td>Console Mgs.</td>
<td>CONSOLE</td>
</tr>
<tr>
<td>Alarm Program</td>
<td>ALARMPROGRAM</td>
</tr>
<tr>
<td>Dump Shared Memory</td>
<td>DUMPSHMEM</td>
</tr>
<tr>
<td>Dump Core</td>
<td>DUMPQCORE</td>
</tr>
<tr>
<td>Dump Core</td>
<td>DUMPQCORE</td>
</tr>
<tr>
<td>Dump Count</td>
<td>DUMPQNUM</td>
</tr>
<tr>
<td>Dump Directory</td>
<td>DUMPDIR</td>
</tr>
</tbody>
</table>

*Figure 12-5. Diagnostics Screen with Parameter Names*

Figure 12-6 shows which ONCONFIG parameters correspond to the PDQ screen.

**PARALLEL DATABASE QUERIES PARAMETERS**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>ONCONFIG Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max PDQ Priority</td>
<td>MAX_PDQPRIORITY</td>
</tr>
<tr>
<td>Decision Support Queries</td>
<td>DS_MAX_QUERIES</td>
</tr>
<tr>
<td>Decision Support Memory (Kbytes)</td>
<td>DS_TOTAL_MEMORY</td>
</tr>
<tr>
<td>Maximum Decision Support Scans</td>
<td>DS_MAX_SCANS</td>
</tr>
<tr>
<td>Dataskip</td>
<td>DATASKIP</td>
</tr>
<tr>
<td>Optimizer Hint</td>
<td>OPTCOMPIND</td>
</tr>
<tr>
<td>Non PDQ Memory</td>
<td>DS_NONPDQ_QUERY_MEM</td>
</tr>
</tbody>
</table>

*Figure 12-6. PDQ Screen with Parameter Names*

Figure 12-7 shows the ON-Monitor screen for creating a dbspace.

**CREATE DBSPACE**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>ONCONFIG Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dbspace Name</td>
<td></td>
</tr>
<tr>
<td>Page Size</td>
<td>2 Kbytes</td>
</tr>
<tr>
<td>Primary Chunk Information:</td>
<td></td>
</tr>
<tr>
<td>Full Pathname</td>
<td></td>
</tr>
<tr>
<td>Offset</td>
<td>0 Kbytes</td>
</tr>
<tr>
<td>Size</td>
<td>0 Kbytes</td>
</tr>
<tr>
<td>Mirror</td>
<td></td>
</tr>
<tr>
<td>Temp</td>
<td></td>
</tr>
<tr>
<td>Mirror Chunk Information:</td>
<td></td>
</tr>
<tr>
<td>Full Pathname</td>
<td></td>
</tr>
<tr>
<td>Offset</td>
<td>0 Kbytes</td>
</tr>
</tbody>
</table>

*Figure 12-7. Create Dbspace Screen*

**Note:** All tables, indexes, and other allocations within the dbspace use pages of the specified page size. The value for Page Size must be a multiple of the page size of the root dbspace.
Chapter 13. The onparams Utility

In This Chapter

- onparams Syntax ................................................................. 13-2
- onparams -a -d dbspace: Add a logical-log file .......................... 13-2
- onparams -d -l lognum: Drop a logical-log file ....................... 13-3
- onparams -p: Change physical-log parameters .......................... 13-3
  - Changing Up After You Change the Physical-Log Size or Location ... 13-4
  - Changing the Size of the Physical Log and Using Non-Default Page Sizes ... 13-4
  - Using a Text Editor to Change the Physical-Log Size or Location ........ 13-4
- onparams -b: Add a new buffer pool ...................................... 13-4
- Examples of onparams Commands ......................................... 13-6

In This Chapter

This chapter shows you how to use the following onparams options:

- "onparams -a -d dbspace: Add a logical-log file" on page 13-2
- "onparams -d -l lognum: Drop a logical-log file" on page 13-3
- "onparams -p: Change physical-log parameters" on page 13-3
- "onparams -b: Add a new buffer pool" on page 13-4

Any onparams command fails if a storage-space backup is in progress. If you do not use any options, onparams returns a usage statement.

In addition to using onparams to manage your database server, you can also use Administrative API commands. Table 13-1 on page 13-1 identifies the onparams options and the Administrative API command string.

Table 13-1. onparams Options

<table>
<thead>
<tr>
<th>Function</th>
<th>onparams Command</th>
<th>Database Server Mode</th>
<th>Administrative API command string</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a logical-log file</td>
<td>onparams -a -d dbspace [-i]</td>
<td>Online, quiescent, or fast-recovery mode</td>
<td>ADD LOG</td>
</tr>
<tr>
<td>Drop a logical-log file</td>
<td>onparams -d -l lognum</td>
<td>Online, quiescent, or fast-recovery mode</td>
<td>DROP LOG</td>
</tr>
<tr>
<td>Change the size or location of</td>
<td>onparams -p</td>
<td>Quiescent mode only</td>
<td>ALTER PLOG</td>
</tr>
<tr>
<td>the physical log</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a new buffer pool</td>
<td>onparams -b</td>
<td>Online, quiescent, or administration mode</td>
<td>ADD BUFFERPOOL</td>
</tr>
</tbody>
</table>

On UNIX, you must be logged in as user root or user informix to execute onparams. Only user informix is allowed to execute the Administrative API command strings.

On Windows, you must be a member of the Informix-Admin group to execute onparams.
### onparams Syntax

```plaintext
onparams
   -a -d dbspace
   -d -1 lognum
   -p
   -b
   -V
   -version
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-V</td>
<td>Displays the software version number and the serial number</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
<tr>
<td>-version</td>
<td>Displays the build version, host, OS, number and date, as well as the GLS version</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
</tbody>
</table>

### onparams -a -d dbspace: Add a logical-log file

**Syntax:**

```plaintext
onparams -a -d dbspace [-i] [-s size]
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a -d dbspace</td>
<td>Adds a logical-log file to the end of the log-file list to the specified dbspace</td>
<td>You can add a log file to a dbspace only if the database server has adequate contiguous space. The newly added log files have a status of A and are immediately available for use. You can add a log file during a backup. You can have a maximum of 32,767 logical-log files. Use onstat -l to view the status of your logical-log files. It is recommended that you take a level-0 backup of the root dbspace and the dbspace that contains the log file as soon as possible. You cannot add a log file to a blobspace or abspace. Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>-i</td>
<td>Inserts the logical-log file after the current log file</td>
<td>Use this option when the Log File Required alarm prompts you to add a logical-log file.</td>
</tr>
<tr>
<td>-s size</td>
<td>Specifies a size in kilobytes for the new logical-log file</td>
<td>This value must be an unsigned integer greater than or equal to 200 kilobytes. If you do not specify a size with the -s option, the size of the log file is taken from the value of the LOGSIZE parameter in the ONCONFIG file when database server disk space was initialized. For information on changing LOGSIZE, see the chapter on managing logical-log files in the IBM Informix Administrator’s Guide.</td>
</tr>
</tbody>
</table>

Use ADD CHUNK as the Administrative API command string for `onparams -a -d dbspace [-i]`. 
onparams -d -l lognum: Drop a logical-log file

Syntax:

```
$ onparams -d -l lognum [-y]
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d -l lognum</td>
<td>Allows you to drop a logical-log file specified by the log file number.</td>
<td><strong>Restrictions:</strong> This value must be an unsigned integer greater than or equal to 0. The database server requires a minimum of three logical-log files at all times. You cannot drop a log file if the database server is configured for three logical-log files. Drop log files one at a time. <strong>Additional Information:</strong> You can obtain the <code>lognum</code> from the <code>number</code> field of <code>onstat -l</code>. The sequence of <code>lognum</code> might be out of order. You can drop a log file immediately that has a status of newly Added (A). If you drop a log file that has a status of Used (U) or Free (F), the database server marks it as Deleted (D) and drops it when you take a level-0 backup of all the dbspaces.</td>
</tr>
<tr>
<td>-y</td>
<td>Causes the database server to automatically respond yes to all prompts</td>
<td>None.</td>
</tr>
</tbody>
</table>

Use DROP LOG as the Administrative API command string for `onparams -d -l lognum`.

When you move logical-log files to another dbspace, use the `onparams` commands to add and drop logical-log files. See moving a logical-log file, in the chapter on managing logical-log files in the IBM Informix Administrator’s Guide.

onparams -p: Change physical-log parameters

Syntax:

```
$ onparams -p [-s size] [-d dbspace [-y]
```

Notes:
1. Only one occurrence of this item is allowed

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-p</td>
<td>Changes the location or size of the physical log</td>
<td>You can use <code>onparams -p</code> with <code>-s</code>, <code>-d</code>, or both. The database server must be online; the server does not need to be restarted for the changes take effect.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>---------------------</td>
</tr>
<tr>
<td>-d dbspace</td>
<td>Changes the location of the physical log to the specified dbspace</td>
<td>The space allocated for the physical log must be contiguous. Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>-s size</td>
<td>Changes the size (in kilobytes) of the physical log</td>
<td>This value must be an unsigned integer greater than or equal to 200 kilobytes. <strong>Warning:</strong> If you move the log to a dbspace without adequate contiguous space or increase the log size beyond the available contiguous space, the operation will fail and the physical log will not change.</td>
</tr>
<tr>
<td>-y</td>
<td>Causes the database server to automatically respond yes to all prompts</td>
<td>None.</td>
</tr>
</tbody>
</table>

**Back Up After You Change the Physical-Log Size or Location**

Changes to the physical log do not take effect until you restart the database server. To restart the database server immediately, execute the `onparams` command with the `-y` option.

Create a level-0 backup of the root dbspace immediately after you restart the database server. This backup is critical for proper recovery of the database server.

**Changing the Size of the Physical Log and Using Non-Default Page Sizes**

If you use non-default page sizes, you might need to increase the size of your physical log. If you perform many updates to non-default pages you might need a 150 to 200 percent increase of the physical log size. Some experimentation might be needed to tune the physical log. You can adjust the size of the physical log as necessary according to how frequently the filling of the physical log triggers checkpoints.

**Using a Text Editor to Change the Physical-Log Size or Location**

Another way to change the size or location of the physical-log is to edit the ONCONFIG file and restart the database server. For information on changing the physical-log location and size, see the chapter on managing the physical-log in the *IBM Informix Administrator’s Guide*.

Use ALTER PLOG as the Administrative API **command** string for `onparams -p`.

**onparams -b: Add a new buffer pool**

**Syntax:**

```
  `onparams -b g size -n number -r number -x percentage`
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-b</td>
<td>Creates a new buffer pool</td>
<td>You can add a new buffer pool while the database server is running. For more information on buffer pools, see the description of the configuration parameter &quot;BUFFERPOOL&quot; on page 1-18 and the information on buffer pools in the IBM Informix Administrator's Guide.</td>
</tr>
<tr>
<td>-g size</td>
<td>Specifies the size in kilobytes of the buffer pages to create</td>
<td>Each dbspace you create with a non-default page size must have a corresponding buffer pool with the corresponding page size. If you create a dbspace with a page size that has no buffer pool, the system will automatically create a buffer pool using the fields in the default line of the BUFFERPOOL parameter. The size of the buffer pages must be between 2 and 16 kilobytes and it must be a multiple of the default page size.</td>
</tr>
<tr>
<td>-m percent</td>
<td>Specifies the percentage of modified pages in the LRU queues at which page cleaning is no longer mandatory</td>
<td>Fractional values are allowed. If you do not specify this option, the percentage used is the value of the lru_min_dirty field as set in the default line of the BUFFERPOOL configuration parameter. For the range of values, see &quot;The lru_min_dirty Field&quot; on page 1-21.</td>
</tr>
<tr>
<td>-n number</td>
<td>Specifies the number of buffers in the buffer pool</td>
<td>If you do not specify this option, the number used is the value of buffers as set in the default line of the BUFFERPOOL configuration parameter. For the range of values, see &quot;The buffers Field&quot; on page 1-20.</td>
</tr>
<tr>
<td>-r number</td>
<td>Specifies the number of LRU (least-recently-used) queues in the shared-memory buffer pool</td>
<td>If you do not include this option, the number of LRU queues allocated is equal to the value of irus as set in the default line of the BUFFERPOOL configuration parameter. For the range of values, see &quot;The irus Field&quot; on page 1-20.</td>
</tr>
<tr>
<td>-x percent</td>
<td>Specifies the default percentage of modified pages in the LRU queues at which the queue is cleaned</td>
<td>Fractional values are allowed. If you do not specify this option, the percentage used is the value of lru_max_dirty as set in the default line of the BUFFERPOOL configuration parameter. For the range of values, see &quot;The lru_max_dirty Field&quot; on page 1-22.</td>
</tr>
</tbody>
</table>

Create a buffer pool that corresponds to the page size of the dbspace. It is recommended that you do this before you create the dbspace. You cannot reduce or increase the number of buffers in an existing buffer pool while the database server is running. You also cannot drop a buffer pool while the database server is running. You can, however, add new buffer pools with a new size while the database server is running.

Buffer pools added with the onparams utility are put into virtual memory, not into resident memory. Upon restart, buffer pool entries will go into resident memory depending on the amount of memory that is available.
When you add a new buffer pool with the onparams utility or when you add a dbspace with a different page size (with the onspaces utility), the settings for the BUFFERPOOL configuration parameter in the ONCONFIG file are rewritten to reflect the new entry.

Use ADD BUFFERPOOL as the Administrative API command string for onparams -b.

Examples of onparams Commands

The following are examples of onparams commands:

- `onparams -a -d rootdbs -s 1000`  # adds a 1000-KB log file to rootdbs
- `onparams -a -d rootdbs -i`       # inserts the log file after the current log
- `onparams -d -l 7`               # drops log 7
- `onparams -p -d dbspace1 -s 3000` # resizes and moves physical-log to dbspace1
- `onparams -b -g 6 -n 3000 -r 2 -x 2.0 -m 1.0`  # adds 3000 buffers of size 6K bytes each with 2 LRUS with maximum dirty of 2% and minimum dirty of 1%
Chapter 14. The onspaces Utility

In This Chapter

- onspaces Syntax ................................................. 14-2
- Administrative API command string .......................... 14-2
- onspaces -a: Add a chunk to a dbspace or blobspace .... 14-3
- onspaces -a: Add a chunk to an sbspace ................. 14-4
- onspaces -b: Create a blobspace .............................. 14-6
- onspaces -d: Create a dbspace ................................. 14-8

Creating a Temporary Dbspace with the -t Option ............... 14-10
Specifying First and Next Extent Size for the tblspace tblspace . 14-11
Specifying a Non-Default Page Size with the Same Size as the Buffer Pool . 14-11

onspaces -s: Create an sbspace ...................... 14-11
- Creating a Temporary Sbspace with the -t Option ........ 14-13
- Creating an Sbspace with the -Df option .................. 14-13
- Changing the -Df Settings ................................. 14-16
- Using the onspaces -g Option .............................. 14-16

- onspaces -x: Create an extspace .......................... 14-16
- onspaces -ch: Change sbspace default specifications .... 14-17
- onspaces -cl: Clean up stray smart large objects in sbspaces . 14-18
- onspaces -d: Drop a chunk in a dbspace, blobspace, or sbspace . 14-19
- onspaces -d: Drop a blobspace, dbspace, extspace, or sbspa ce . 14-20
- onspaces -f: Specify DATASKIP parameter ............... 14-21
- onspaces -m: Start mirroring, .............................. 14-22
- Using a File to Specify Chunk-Location Information with the -f Option . 14-23
- onspaces -r: Stop mirroring .................................. 14-24
- onspaces -ren: Rename a dbspace, blobspace, sbspace, or extspace . 14-24
- Renaming a Dbspace, Blobspace, Sbspace, or Extspace when Enterprise Replication Is Active . 14-25
- Performing an Archive after Renaming a Space .......... 14-25
- onspaces -s: Change status of a mirrored chunk ......... 14-26

This chapter shows you how to use the following onspaces options:

- “onspaces -a: Add a chunk to a dbspace or blobspace” on page 14-3
- “onspaces -a: Add a chunk to an sbspace” on page 14-4
- “onspaces -b: Create a blobspace” on page 14-6
- “onspaces -d: Create a dbspace” on page 14-8
- “onspaces -s: Create an sbspace” on page 14-11
- “onspaces -x: Create an extspace” on page 14-16
- “onspaces -ch: Change sbspace default specifications” on page 14-17
- “onspaces -cl: Clean up stray smart large objects in sbspaces” on page 14-18
- “onspaces -d: Drop a chunk in a dbspace, blobspace, or sbspace” on page 14-19
- “onspaces -d: Drop a blobspace, dbspace, extspace, or sbspace” on page 14-20
- “onspaces -f: Specify DATASKIP parameter” on page 14-21
- “onspaces -m: Start mirroring” on page 14-22
- “onspaces -r: Stop mirroring” on page 14-24
- “onspaces -ren: Rename a dbspace, blobspace, sbspace, or extspace” on page 14-24
- “onspaces -s: Change status of a mirrored chunk” on page 14-26
When you use onspaces or ISA to manage a storage space, the database server updates information about the space in the `oncfg_servername.servernum` file. For more information on the `oncfg` file, refer to Appendix A, “Files That the Database Server Uses,” on page A-1.

You can specify a maximum of 2047 chunks for a storage space, and a maximum of 2047 storage spaces on the database server system. The storage spaces can be any combination of dbspaces, blobspaces, and sbspaces.

On UNIX, you must be logged in as user root or user informix to execute `onspaces`. On Windows, you must be a member of the Informix-Admin group.

### onspaces Syntax

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-V</td>
<td>Displays the software version number and the serial number</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
<tr>
<td>-version</td>
<td>Displays the build version, host, OS, number and date, as well as the GLS version</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
</tbody>
</table>

### Administrative API command string

The following onspaces options have equivalent Administrative API command strings:

<table>
<thead>
<tr>
<th>onspaces option</th>
<th>Administrative API command string</th>
</tr>
</thead>
<tbody>
<tr>
<td>onspaces -a</td>
<td>ADD CHUNK CREATE CHUNK</td>
</tr>
<tr>
<td>onspaces -c -b</td>
<td>CREATE BLOBSpace</td>
</tr>
<tr>
<td>onspaces -c -d</td>
<td>CREATE DBSPACE</td>
</tr>
<tr>
<td>onspaces -c -d -t</td>
<td>CREATE TEMPDDBSPACE</td>
</tr>
<tr>
<td>onspaces -c -S</td>
<td>CREATE SBSPACE</td>
</tr>
<tr>
<td>onspaces -cl</td>
<td>CLEAN SBSPACE</td>
</tr>
</tbody>
</table>
**onspaces option** | **Administrative API command string**
---|---
**onspaces -d** | DROP BLOBSPACE  
DROP CHUNK  
DROP DBSPACE  
DROP LOG  
DROP SBSPACE  
DROP TEMPDBSPACE
**onspaces -d -l** | DROP LOG
**onspaces -Df** | SET SBSPACE ACCESSTIME ON  
SET SBSPACE ACCESSTIME OFF  
SET SBSPACE AVG_LO_SIZE  
SET SBSPACE LOGGING ON  
SET SBSPACE LOGGING OFF
**onspaces -f** | SET DATASKIP ON  
SET DATASKIP OFF
**onspaces -m** | START MIRIRRING
**onspaces -r** | STOP MIRIRRING
**onspaces -ren** | RENAME SPACE
**onspaces -s** | SET CHUNK ONLINE  
SET CHUNK OFFLINE

---

**onspaces -a: Add a chunk to a dbspace or blobspace**

**Syntax:**

```
  onspaces -a <dbspace> -p <pathname> <o> <offset>
  onspaces -a <blobspace> -p \.<drive> <o> <offset>
  onspaces -s <size> <m> <pathname> <o> <offset>
  onspaces -s <size> \.<drive> <o> <offset>
```

**Notes:**

1. Windows only

Use **onspaces -a** to add a chunk to a dbspace or blobspace.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>Indicates that a chunk is to be added</td>
<td>A dbspace, blobspace, or sbspace can contain up to 32,766 chunks.</td>
</tr>
</tbody>
</table>
| **drive** | Specifies the Windows drive to allocate as unbuffered board space  
The format can be either  
\.<drive>, where drive is the drive letter assigned to a disk partition, or  
\.<PhysicalDrive<number>>, where PhysicalDrive is a constant value and number is the physical drive number. | For more information on allocating unbuffered disk space, see allocating raw disk space on Windows in the chapter on managing disk space in the *IBM Informix Administrator’s Guide*. Example:  
\.<drive>
For pathnamen syntax, see your operating-system documentation. |
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-m pathname</td>
<td>Specifies an optional pathname and offset to the chunk that mirrors the new chunk. Also see the entries for pathname and offset in this table.</td>
<td>For more information, see adding a chunk to a dbspace and adding a chunk to a blobspace, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>offset</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-o offset</td>
<td>After the -a option, offset indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the new blobspace or dbspace</td>
<td>Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes. For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-p pathname</td>
<td>Indicates the disk partition or unbuffered device of the initial chunk of the blobspace or dbspace that you are adding. The chunk must be an existing unbuffered device or buffered file.</td>
<td>The chunk name can be up to 128 bytes. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. UNIX example (unbuffered device): /dev/rdsk/c0t3d0s4 UNIX example (buffered device): /ix/ids9.2/dblchunk Windows example: c:\ifmdata\ol_icecream\mychunk1.dat For pathname syntax, see your operating-system documentation.</td>
</tr>
<tr>
<td>-s size</td>
<td>Indicates, in kilobytes, the size of the new blobspace or dbspace chunk</td>
<td>Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes.</td>
</tr>
<tr>
<td>blobspace</td>
<td>Names the blobspace to which you are adding a chunk</td>
<td>See adding a chunk to a blobspace in the chapter on managing disk space in the IBM Informix Administrator’s Guide. Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>dbspace</td>
<td>Names the dbspace to which you are adding a chunk</td>
<td>See adding a chunk to a dbspace, in the chapter on managing disk space in the IBM Informix Administrator’s Guide. Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax.</td>
</tr>
</tbody>
</table>

Use ADD CHUNK or CREATE CHUNK as the Administrative API command string for onspaces -a.

**onspaces -a: Add a chunk to an sbspace**

**Syntax:**

```plaintext
onspaces -a sbspace -p pathname -o offset -s size
```
Use **onspaces -a** to add a chunk to an sbspace.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>Indicates that a chunk is to be added</td>
<td>An sbspace can contain up to 32,766 chunks.</td>
</tr>
<tr>
<td>-m pathname offset</td>
<td>Specifies an optional pathname and offset to the chunk that mirrors the new chunk. Also see the entries for pathname and offset in this table.</td>
<td>For background information, see adding a chunk to an sbspace, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-Mo mdoffset</td>
<td>Indicates, in kilobytes, the offset into the disk partition or into the device where metadata should be stored</td>
<td>Value can be an integer between 0 and the chunk size. You cannot specify an offset that causes the end of the metadata space to be past the end of the chunk. For background information, see sizing sbspace metadata, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-Ms mdsize</td>
<td>Specifies the size, in kilobytes, of the metadata area allocated in the initial chunk. The remainder is user-data space</td>
<td>Value can be an integer between 0 and the chunk size. For background information, see sizing sbspace metadata, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-o offset</td>
<td>After the -a option, offset indicates, in kilobytes, the offset into the disk partition or into the unbuffered device to reach the initial chunk of the new sbspace or dspace.</td>
<td>Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 2 or 4 gigabytes, depending on the platform. For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-p pathname</td>
<td>Indicates the disk partition or unbuffered device of the initial chunk of the sbspace that you are creating. The chunk must be an existing unbuffered device or buffered file.</td>
<td>The chunk name can be up to 128 bytes. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. For pathname syntax, see your operating-system documentation.</td>
</tr>
<tr>
<td>-U</td>
<td>Specifies that the entire chunk should be used to store user data</td>
<td>The -M and -U options are mutually exclusive. For background information, see adding a chunk to an sbspace, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-s size</td>
<td>Indicates, in kilobytes, the size of the new sbspace chunk</td>
<td>Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes.</td>
</tr>
</tbody>
</table>
Use ADD CHUNK or CREATE CHUNK as the Administrative API command string for onspaces -a.

### onspaces -c -b: Create a blobspace

**Syntax:**

```plaintext
onspaces -c -b blobspace -g pageunit -p pathname -o offset -s size
```

```
-m pathname offset
-m \--\--\--\--\-- drive offset
```

**Notes:**

1. Windows Only

Use onspaces -c -b to create a blobspace.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-b blobspace</td>
<td>Names the blobspace to be created</td>
<td>The blobspace name must be unique and cannot exceed 128 bytes. It must begin with a letter or underscore and must contain only letters, numbers, underscores, or the $ character. For more information, see creating a blobspace, in the chapter on managing disk space in the IBM Informix Administrator’s Guide. The syntax must conform to the Identifier segment. For more information, see the IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>-c</td>
<td>Creates a dbspace, blobspace, sbspace, or extspace</td>
<td>After you create a storage space, you must back up both this storage space and the root dbspace. If you create a storage space with the same name as a deleted storage space, perform another level-0 backup to ensure that future restores do not confuse the new storage space with the old one. For more information, see creating a dbspace, blobspace, or extspace, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>drive</td>
<td>Specifies the Windows drive to allocate as unbuffered disk space. The format can be either \&lt;drive&gt;, where drive is the drive letter assigned to a disk partition, or \PhysicalDrive&lt;number&gt;, where PhysicalDrive is a constant value and number is the physical drive number.</td>
<td>For information on allocating unbuffered disk space, see allocating unbuffered disk space on Windows in the chapter on managing disk space in the IBM Informix Administrator’s Guide. Examples: \F: \PhysicalDrive2</td>
</tr>
<tr>
<td>-g pageunit</td>
<td>Specifies the blobspace blobpage size in terms of page unit, the number of disk pages per blobpage</td>
<td>Unsigned integer. Value must be greater than 0. For more information, see blobpage size considerations, in the chapter on I/O Activity in the IBM Informix Performance Guide.</td>
</tr>
<tr>
<td>-m pathname offset</td>
<td>Specifies an optional pathname and offset to the chunk that mirrors the initial chunk of the new blobspace or dbspace. Also see the entries for -p pathname and -o offset in this table.</td>
<td>For more information, see creating a dbspace or a blobspace in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-o offset</td>
<td>Indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the new blobspace, dbspace, or sbspace</td>
<td>Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 2 or 4 gigabytes, depending on the platform. For more information, see allocating raw disk space, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-p pathname</td>
<td>Indicates the disk partition or device of the initial chunk of the blobspace or dbspace that you are creating.</td>
<td>The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. UNIX example (unbuffered device): /dev/rdsk/c0t3d0s4 UNIX example (buffered device): /ix/ids9.2/db1chunk Windows example: c:\Ifmxdata\ol_icecream\mychunk1.dat For pathname syntax, see your operating-system documentation.</td>
</tr>
<tr>
<td>-s size</td>
<td>Indicates, in kilobytes, the size of the initial chunk of the new blobspace or dbspace</td>
<td>Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum chunk size is 2 or 4 terabytes, depending on the platform.</td>
</tr>
</tbody>
</table>

Use CREATE BLOBSPACE as the Administrative API command string for onspaces -c -b.
onspaces -c -d: Create a dbspace

Syntax:

```
/onspaces -c -d dbspace -p pathname -o offset -s size
```

```
-ef extentsize -en extentsize
-t [m] pathname offset (1) -k pagesize
```

Notes:

1. Windows Only

Use `onspaces -c -d` to create a dbspace or a temporary dbspace.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-c</td>
<td>Creates a dbspace</td>
<td>After you create a storage space, you must back up both this storage space and the root dbspace. If you create a storage space with the same name as a deleted storage space, perform another level-0 backup to ensure that future restores do not confuse the new storage space with the old one. For more information, see creating a dbspace in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td></td>
<td>You can create up to 2047 storage spaces of any type.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The syntax must conform to the Identifier segment. For more information, see the <em>IBM Informix Guide to SQL: Syntax</em>.</td>
</tr>
<tr>
<td>drive</td>
<td>Specifies the Windows drive to allocate as unbuffered disk space</td>
<td>For information on allocating unbuffered disk space, see allocating unbuffered disk space on Windows in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td></td>
<td>The format can be either <code>\.&lt;drive&gt;</code>, where drive is the drive letter assigned to a disk partition, or <code>\.&lt;PhysicalDrive&lt;number&gt;&gt;</code>, where PhysicalDrive is a constant value and number is the physical drive number.</td>
<td>Examples:</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>\.&lt;drive&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>\.&lt;PhysicalDrive&lt;number&gt;&gt;</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td>For pathnames, see your operating-system documentation.</td>
</tr>
<tr>
<td>-d dbspace</td>
<td>Names the dbspace to be created</td>
<td>The dbspace name must be unique and cannot exceed 128 bytes. It must begin with a letter or underscore and must contain only letters, numbers, underscores, or the $ character. For more information, see creating a dbspace, in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>. The syntax must conform to the Identifier segment. For more information, see the <em>IBM Informix Guide to SQL: Syntax</em>.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>-ef extentsize</code></td>
<td>Indicates, in kilobytes, the size of the next extents in the tblspace tblspace</td>
<td>The minimum size of the next extents for the tblspace tblspace of a non-root dbspace is equivalent to 4 dbspace pages, specified in K. For example: 8 KB for a 2 KB page size dbspace, 16 KB for a 4 KB page size dbspace, 32 KB for an 8 KB page size dbspace. The default size for a next extent is 50 dbspace pages. The maximum size of a tblspace tblspace extent is 1048572 pages. On a 2 KB pagesize system this would evaluate to approximately 2 GB. If there is not enough space for a next extent in the primary chunk, the extent is allocated from another chunk. If the specified space is not available, the closest available space is allocated. For more information, see specifying first and next extent size in the chapter on managing dbspaces in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td><code>-en extentsize</code></td>
<td>Indicates, in kilobytes, the size of the first extent for the tblspace tblspace</td>
<td>The minimum, and default, size of the first extent for the tblspace tblspace of a non-root dbspace is equivalent to 50 dbspace pages, specified in K. For example: 100 KB for a 2 KB page size dbspace, 200 KB for a 4 KB page size dbspace, 400 KB for an 8 KB page size dbspace. The maximum size of a tblspace tblspace extent is 1048575 pages minus the space needed for any system objects. On a 2 KB pagesize system this would evaluate to approximately 2 GB. For more information, see specifying first and next extent size in the chapter on managing dbspaces in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td><code>-k pagesize</code></td>
<td>Indicates in kilobytes, the non-default page size for the new dbspace. For systems with sufficient storage, performance advantages of a larger page size can include the following: • Reduced depth of B-tree indexes, even for smaller index keys • You can group on the same page long rows that currently span multiple pages of the default page size • Checkpoint time is typically reduced with larger pages • You can define a different page size for temporary tables, so that they have a separate buffer pool.</td>
<td>The page size must be between 2KB and 16KB and must be a multiple of the default page size. For example, if the default page size is 2KB, then <code>pagesize</code> can be 2, 4, 6, 8, 10, 12, 14, or 16. If the default page size is 4KB (Windows), then <code>pagesize</code> can be 4, 8, 12, or 16. For more information, see creating a dbspace with a non-default page size in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>-m pathname</code></td>
<td>Specifies an optional pathname and offset to the chunk that mirrors the initial chunk of the new dbspace. Also see the entries for <code>-p pathname</code> and <code>-o offset</code> in this table.</td>
<td>For more information, see creating a dbspace in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td><code>-o offset</code></td>
<td>Indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the new dbspace.</td>
<td>Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The offset must be a multiple of the page size. The maximum offset is 2 or 4 gigabytes, depending on the platform. For more information, see allocating raw disk space, in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td><code>-p pathname</code></td>
<td>Indicates the disk partition or device of the initial chunk of the dbspace that you are creating.</td>
<td>The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. UNIX example (unbuffered device): <code>/dev/rdsk/c0t3d0s4</code> UNIX example (buffered device): <code>/ix/ids9.2/db1chunkWindows example:c:\Ifmxdata\01_icecream\mychunk1.dat</code> For pathname syntax, see your operating-system documentation.</td>
</tr>
<tr>
<td><code>-s size</code></td>
<td>Indicates, in kilobytes, the size of the initial chunk of the new dbspace.</td>
<td>Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum chunk size is 2 or 4 terabytes, depending on the platform.</td>
</tr>
<tr>
<td><code>-t</code></td>
<td>Creates a temporary dbspace for storage of temporary tables.</td>
<td>You cannot mirror a temporary dbspace. You cannot specify the first and next extent sizes for the tblspace tblspace of a temporary dbspace. For more information, see temporary dbspaces, in the chapter on data storage, and creating a temporary dbspace, in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
</tbody>
</table>

Use CREATE DBSPACE as the Administrative API *command* string for **onspaces** -c -d.

**Creating a Temporary Dbspace with the -t Option**

When you create a temporary dbspace with **onspaces**, the database server uses the newly created temporary dbspace, after you perform the following steps:

- Add the name of the new temporary dbspace to your list of temporary dbspaces in the DBSPACETEMP configuration parameter, the DBSPACETEMP environment variable, or both.
• Restart the database server.

Use CREATE TEMPDBSPACE as the Administrative API command string for onspaces -c -d -t.

Specifying First and Next Extent Size for the tblspace tblspace
You cannot specify the first and next extent of a temporary dbspace. The extent size for temporary dbspaces is 100 kilobytes for a 2 kilobyte page system or 200 kilobytes for a 4 kilobyte page system.

To specify the first and next extent sizes of a root tblspace, use the TBLTBLFIRST and TBLTBLNEXT configuration parameters before you create the root dbspace the first time you start the database server.

Specifying a Non-Default Page Size with the Same Size as the Buffer Pool
When you create a dbspace with a non-default page size, you must also create a buffer pool specific to that page size. It is recommended that you create the buffer pool before you create the dbspace. Use the onparams utility to create a buffer pool. For more information, see onparams -b: Add a new buffer pool on page 13-4.

When you add a dbspace with a different page size with the onspaces utility or you add a new buffer pool (with the onparams utility), a new BUFFERPOOL line is appended in the BUFFERPOOL configuration parameter in the ONCONFIG file to reflect the new entry and it is rewritten to disk.

Notes:
1. You cannot change the page size of a dbspace after you create it.
2. You cannot store logical or physical logs in a dbspace that is not the default platform page size.
3. If a dbspace is created when a buffer pool with that page size does not exist, Dynamic Server creates a buffer pool using the values of the fields of the default line of the BUFFERPOOL parameter. You cannot have multiple buffer pools with the same page size.

onspaces -c -S: Create an sbspace

Syntax:

```
>>onspaces -c -S sbspace -t p pathname -o offset -s size
```

Use onspaces -c -S to create a sbspace or a temporary sbspace.
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-S sbspace</td>
<td>Names the sbspace to be created</td>
<td>The sbspace name must be unique and must not exceed 128 bytes. It must begin with a letter or underscore and must contain only letters, numbers, underscores, or the $ character. Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>-c</td>
<td>Creates an sbspace</td>
<td>None.</td>
</tr>
<tr>
<td>-m pathname offset</td>
<td>Specifies an optional pathname and offset to the chunk that mirrors the initial chunk of the new sbspace Also see the entries for -p pathname and -o offset in this table.</td>
<td>For more information, see sbspaces in the chapter on data storage, and creating an sbspace, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-Mo mdoffset</td>
<td>Indicates, in kilobytes, the offset into the disk partition or into the device where metadata will be stored.</td>
<td>Restrictions: Value can be an integer between 0 and the chunk size. You cannot specify an offset that causes the end of the metadata space to be past the end of the chunk. References: For more information, see sizing sbspace metadata, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-Ms mdsize</td>
<td>Specifies the size, in kilobytes, of the metadata area allocated in the initial chunk The remainder is user-data space.</td>
<td>Restrictions: Value can be an integer between 0 and the chunk size.</td>
</tr>
<tr>
<td>-o offset</td>
<td>Indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the sbspace</td>
<td>Restrictions: Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum chunk size is 4 terabytes for systems with a two-kilobyte page size and 8 terabytes for systems with a four-kilobyte page size. References: For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-p pathname</td>
<td>Indicates the disk partition or unbuffered device of the initial chunk of the sbspace</td>
<td>The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. References: For pathname syntax, see your operating-system documentation.</td>
</tr>
<tr>
<td>-s size</td>
<td>Indicates, in kilobytes, the size of the initial chunk of the new sbspace</td>
<td>Restrictions: Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum chunk size is 2 or 4 gigabytes, depending on the platform.</td>
</tr>
</tbody>
</table>
-t

Creates a temporary sbspace for storage of temporary smart large objects. You can specify the size and offset of the metadata area.

Key Considerations:

Restrictions: You cannot mirror a temporary sbspace. You can specify any -Df option, except the LOGGING=ON option, which has no effect.

References: For more information, see "Creating a Temporary Sbspace with the -t Option" on page 14-13.

-Df default list

Lists default specifications for smart large objects stored in the sbspace.

Key Considerations:

Restrictions: Tags are separated by commas. If a tag is not present, system defaults take precedence. The list must be enclosed in double quotation marks (" ) on the command line.

References: For a list of tags and their parameters, see Table 14-1 on page 14-13.

Creating a Temporary Sbspace with the -t Option

This example creates a temporary sbspace of 1000 kilobytes:

```
onspaces -c -S tempsbsp -t -p ./tempsbsp -o 0 -s 1000
```

You can optionally specify the name of the temporary sbspace in the SBSPACETEMP configuration parameter. Restart the database server so that it can use the temporary sbspace.

Creating an Sbspace with the -Df option

When you create an sbspace with the optional -Df option, you can specify several default specifications that affect the behavior of the smart large objects stored in the sbspace. The default specifications must be expressed as a list separated by commas. The list need not contain all of the tags. The list of tags must be enclosed in double quotation marks (" ). The table in Table 14-1 on page 14-13 describes the tags and their default values.

The four levels of inheritance for sbspace characteristics are system, sbspace, column, and smart large objects. For more information, see smart large objects in the chapter on where data is stored in the IBM Informix Administrator’s Guide.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESTIME</td>
<td>ON or OFF</td>
<td>OFF</td>
<td>When set to ON, the database server tracks the time of access to all smart large objects stored in the sbspace. For information about altering storage characteristics of smart large objects, see the IBM Informix DataBlade API Programmer’s Guide.</td>
</tr>
</tbody>
</table>
Table 14-1. -Df Default Specifications (continued)

<table>
<thead>
<tr>
<th>Tag</th>
<th>Values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVG_LO_SIZE</td>
<td>Windows: 4 to 2<strong>31 UNIX: 2 to 2</strong>31</td>
<td>8</td>
<td>Specifies the average size, in kilobytes, of the smart large object stored in the sbspace. The database server uses this value to calculate the size of the metadata area. Do not specify AVG_LO_SIZE and -Ms together. You can specify AVG_LO_SIZE and the metadata offset (-Mo) together. If the size of the smart large object exceeds 2<strong>31, specify 2</strong>31. If the size of the smart large object is less than 2 on UNIX or less than 4 in Windows, specify 2 or 4. Error 131 is returned if you run out of space in the metadata and reserved areas in the sbspace. To allocate additional chunks to the sbspace that consist of metadata area only, use the -Ms option instead. For more information, see creating smart large objects, in the chapter on managing data on disk in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>BUFFERING</td>
<td>ON or OFF</td>
<td>ON</td>
<td>Specifies the buffering mode of smart large objects stored in the sbspace. If set to ON, the database server uses the buffer pool in the resident portion of shared memory for smart-large-object I/O operations. If set to OFF, the database server uses light I/O buffers in the virtual portion of shared memory (lightweight I/O operations). BUFFERING = OFF is incompatible with LOCK_MODE = RANGE and creates a conflict. For more information, see lightweight I/O, in the chapter on configuration effects on memory in the IBM Informix Performance Guide.</td>
</tr>
<tr>
<td>LOCK_MODE</td>
<td>RANGE or BLOB</td>
<td>BLOB</td>
<td>Specifies the locking mode of smart large objects stored in the sbspace. If set to RANGE, only a range of bytes in the smart large object is locked. If set to BLOB, the entire smart large object is locked. LOCK_MODE = RANGE is incompatible with BUFFERING = OFF and creates a conflict. For more information, see smart large objects, in the chapter on locking in the IBM Informix Performance Guide.</td>
</tr>
</tbody>
</table>
Table 14-1. -Df Default Specifications (continued)

<table>
<thead>
<tr>
<th>Tag</th>
<th>Values</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGGING</td>
<td>ON or OFF</td>
<td>OFF</td>
<td>Specifies the logging status of smart large objects stored in the sbspace. If set to ON, the database server logs changes to the user data area of the sbspace. When you turn on logging for an sbspace, take a level-0 backup of the sbspace. When you turn off logging, the following message displays: You are turning off smart large object logging. For more information, see smart large objects, in the chapters on data storage and logging in the IBM Informix Administrator's Guide. For information about onspaces -ch messages, see Appendix E, “Error Messages,” on page E-1.</td>
</tr>
<tr>
<td>EXTENT_SIZE</td>
<td>4 to 2**31</td>
<td>None</td>
<td>Specifies the size, in kilobytes, of the first allocation of disk space for smart large objects stored in the sbspace when you create the table. Let the system select the EXTENT_SIZE value. To reduce the number of extents in a smart large object, use mi_lo_specset_estbytes (DataBlade API) or ifx_lo_specset_estbytes (ESQL/C) to hint to the system the total size of the smart large object. The system attempts to allocate a single extent for the smart large object. For more information, see smart large objects, in the chapter on where data is stored in the IBM Informix Administrator's Guide. For information about altering storage characteristics of smart large objects, see the IBM Informix DataBlade API Programmer’s Guide or the IBM Informix ESQL/C Programmer’s Manual.</td>
</tr>
<tr>
<td>MIN_EXT_SIZE</td>
<td>2 to 2**31</td>
<td>Windows: 4UNIX: 2</td>
<td>Specifies the minimum amount of space, in kilobytes, to allocate for each smart large object. The following message displays: Changing the sbspace minimum extent size: old value value1 new value value2. For information about tuning this value, see smart large objects, in the chapter on configuration effects on I/O utilization in the IBM Informix Performance Guide. For information about onspaces -ch messages, see Appendix E, “Error Messages,” on page E-1.</td>
</tr>
<tr>
<td>NEXT_SIZE</td>
<td>4 to 2**31</td>
<td>None</td>
<td>Specifies the extent size, in kilobytes, of the next allocation of disk space for smart large objects when the initial extent in the sbspace becomes full. Let the system select the NEXT_SIZE value. To reduce the number of extents in a smart large object, use mi_lo_specset_estbytes or ifx_lo_specset_estbytes to hint to the system the total size of the smart large object. The system attempts to allocate a single extent for the smart large object. For more information, see smart large objects, in the chapter on where data is stored in the IBM Informix Administrator’s Guide. For information about obtaining the size of smart large objects, see the IBM Informix DataBlade API Programmer’s Guide or the IBM Informix ESQL/C Programmer’s Manual.</td>
</tr>
</tbody>
</table>
This example creates a 20-megabyte mirrored sbspace, `eg_sbsp`, with the following specifications:

- An offset of 500 kilobytes for the primary and mirror chunks
- An offset of 200 kilobytes for the metadata area
- An average expected smart-large-object size of 32 kilobytes
- Log changes to the smart large objects in the user-data area of the sbspace

```
UNIX Only

% onspaces -c -S eg_sbsp -p /dev/raw_dev1 -o 500 -s 20000
   -m /dev/raw_dev2 500 -Mo 200 -Df "AVG_LO_SIZE=32,LOGGING=ON"
```

### Changing the -Df Settings

As the database server administrator, you can override or change the -Df default settings in one of the following ways:

- To change the default settings for an sbspace, use the `onspaces -ch` option. For more information, refer to ["onspaces -ch: Change sbspace default specifications"](page 14-17)
- To override the following -Df default settings for a specific table, use the SQL statements `CREATE TABLE` or `ALTER TABLE`:
  - LOGGING
  - ACCESSTIME
  - EXTENT_SIZE
  - NEXT_SIZE

For more information on the `ALTER TABLE` and `CREATE TABLE` statements, see the *IBM Informix Guide to SQL: Syntax*.

The programmer can override these -Df default settings with DataBlade API and ESQL/C functions. For information about altering storage characteristics of smart large objects, see the *IBM Informix DataBlade API Programmer’s Guide* and the *IBM Informix ESQL/C Programmer’s Manual*.

### Using the onspaces -g Option

The `onspaces -g` option is not used for sbspaces. The database server uses a different method to determine the number of pages to transfer in an I/O operation for sbspaces than for blobspaces. The database server can automatically determine the block size to transfer in an I/O operation for smart large objects. For more information, see sbspace extent sizes in the chapter on I/O activity in your *IBM Informix Performance Guide*.

Use `CREATE SBSPACE` as the Administrative API command string for `onspaces -c -S`.

---

### onspaces -c -x: Create an extspace

**Syntax:**

```
  onspaces -c -x extspace -l location -o offset -s size
```
Use `onspaces -c -x` to create an extspace.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-c</code></td>
<td>Creates a dbspace, blobspace, sbspace, or extspace</td>
<td>After you create a storage space, you must back up both this storage space and the root dbspace. If you create a storage space with the same name as a deleted storage space, perform another level-0 backup to ensure that future restores do not confuse the new storage space with the old one. For more information, see creating a dbspace, blobspace, or extspace, in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td><code>-l location</code></td>
<td>Specifies the location of the extspace</td>
<td><strong>Restrictions</strong>: String. Value must not be longer than 255 bytes. For more information, see creating an extspace, in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td><code>-o offset</code></td>
<td>Indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the new blobspace, dbspace, or sbspace</td>
<td><strong>Restrictions</strong>: Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 2 or 4 gigabytes, depending on the platform. For more information, see allocating raw disk space, in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td><code>-s size</code></td>
<td>Indicates, in kilobytes, the size of the initial chunk of the new blobspace or dbspace</td>
<td><strong>Restrictions</strong>: Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum chunk size is 2 or 4 terabytes, depending on the platform.</td>
</tr>
<tr>
<td><code>-x extspace</code></td>
<td>Names the extspace to be created</td>
<td><strong>Restrictions</strong>: Extspace names can be up to 128 bytes. They must be unique, begin with a letter or underscore, and contain only letters, digits, underscores, or $ characters. For more information, see extspaces, in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
</tbody>
</table>

**onspaces -ch: Change sbspace default specifications**

**Syntax:**

```
    >>> onspaces -ch sbspace -Df default list
```

Use `onspaces -ch` to change the default specifications of a sbspace.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-ch</code></td>
<td>Indicates that one or more sbspace default specifications are to be changed</td>
<td>None.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>--------------------</td>
</tr>
<tr>
<td><strong>sbspace</strong></td>
<td>Names the sbspace for which to change the default specifications</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For background information, see changing default specifications of an sbspace with <strong>onspaces</strong> in the IBM Informix Performance Guide.</td>
</tr>
</tbody>
</table>
| **-Df default list** | Lists new default specifications for smart large objects stored in the sbspace | Tags are separated by commas. If a tag is not present, system defaults take precedence. The list must be enclosed in double quotation marks (") on the command line.  
For a list of tags and their parameters, see Table 14-1 on page 14-13. |

You can change any of the **-Df** tags with the **onspaces -ch** option. The database server applies the change to each smart large object that was created prior to changing the default specification.

For example, to turn off logging for the sbspace that you created in “Creating an Sbspace with the **-Df** option” on page 14-13, use the following command:

```
onspaces -ch eg_sbsp -Df "LOGGING=OFF"
```

**Note:** After you turn on logging for an sbspace, take a level-0 backup of the sbspace to create a point from which to recover.

---

### onspaces -cl: Clean up stray smart large objects in sbspaces

**Syntax:**

```
onspaces -cl sbspace
```

Use **onspaces -cl** to clean up stray smart large objects in sbspaces.

**Syntax:**

```
—onspaces -cl sbspace—
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-cl</strong></td>
<td>Cleans up stray smart large objects in an sbspace</td>
<td>To find any stray smart large objects, use the <strong>oncheck -ps</strong> command when no users are connected to the database server. The smart large objects with a reference count of 0 are stray objects.</td>
</tr>
<tr>
<td><strong>sbspace</strong></td>
<td>Names the sbspace to be cleaned up</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax.</td>
</tr>
</tbody>
</table>

During normal operation, no unreferenced (stray) smart large objects should exist. When you delete a smart large object, the space is released. If the database server fails or runs out of system memory while you are deleting a smart large object, the smart large object might remain as a stray object.

The following is an example of the **onspaces -cl** command:
onspaces -cl myspace

The best way to find the reference count for a smart large object is to call the mi_lo_stat or ifx_lo_stat functions from a C program. Although the mi_lo_increfcount and mi_lo_decrefcount functions return the reference count, they increment or decrement the reference count. For more information on these functions, see the IBM Informix DataBlade API Function Reference.

Use CLEAN SBSPACE as the Administrative API command string for onspaces -cl.

**onspaces -d: Drop a chunk in a dbspace, blobspace, or sbpace**

**Syntax:**

```
>>> onspaces -d <dbspace> <blobspace> <sbpace> -p <pathname> -o <offset> -f -y
```

Use onspaces -d to drop a chunk in a dbspace, blobspace, or sbpace.

Use DROP CHUNK as the Administrative API command string for onspaces -d.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>Drops a chunk</td>
<td>You can drop a chunk from a dbspace, temporary dbspace, or sbspace when the database server is online or quiescent. For more information, see the chapter on managing disk space in the IBM Informix Administrator’s Guide. You can drop a chunk from a blobspace only when the database server is in quiescent mode.</td>
</tr>
<tr>
<td>-f</td>
<td>Drops an sbspace chunk that contains user data but no metadata. If the chunk contains metadata for the sbspace, you must drop the entire sbspace.</td>
<td>Use the -f option with sbspaces only. If you omit the -f option, you cannot drop an sbspace that contains data. For more information, see dropping a chunk from an sbspace with onspaces, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-o offset</td>
<td>Indicates, in kilobytes, the offset into the disk partition or into the unbuffered device to reach the initial chunk of the dbspace, blobspace, or sbspace that you are dropping</td>
<td>Restrictions: Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes. For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-p pathname</td>
<td>Indicates the disk partition or unbuffered device of the initial chunk of the dbspace, blobspace, or sbspace that you are dropping</td>
<td>The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. For pathname syntax, see your operating-system documentation.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>-------------------</td>
</tr>
<tr>
<td>-y</td>
<td>Causes the database server to automatically respond yes to all prompts</td>
<td>None.</td>
</tr>
<tr>
<td>blobspace</td>
<td>Names the blobspace from which the chunk is dropped</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For more information, see dropping a chunk from a blobspace, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>dbspace</td>
<td>Names the dbspace from which the chunk is dropped</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For more information, see dropping a chunk from a dbspace with onspaces, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>sbspace</td>
<td>Names the sbspace from which the chunk is dropped</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For background information, see dropping a chunk from a dbspace with onspaces, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
</tbody>
</table>

**Important:** You must specify a pathname to indicate to the database server that you are dropping a chunk.

**onspaces -d: Drop a blobspace, dbspace, extspace, or sbspace**

**Syntax:**

```plaintext
onspaces -d [blobspace] [dbspace] [sbspace] [extspace] [-y]
```

Use `onspaces -d` to drop a dbspace, blobspace, sbspace, or extspace.

Use DROP BLOBSPACE, DROP DBSPACE, DROP SBSPACE, or DROP TEMPDBSPACE as the Administrative API command strings for `onspaces -d`.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-d</td>
<td>Indicates that a dbspace, blobspace, sbspace, or extspace is to be dropped</td>
<td>You can drop a dbspace, blobspace, sbspace, or extspace while the database server is online or in quiescent mode. After you drop a storage space, you must back it up to ensure that the sysutils database and the reserved pages are up-to-date. Execute <code>oncheck -pe</code> to verify that no table is currently storing data in the dbspace, blobspace, or sbspace. For more information, see dropping a storage space, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>-y</td>
<td>Causes the database server to automatically respond yes to all prompts</td>
<td>None.</td>
</tr>
</tbody>
</table>
### Database Storage Spaces

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f</td>
<td>Drops an sbspace that contains user data and metadata</td>
<td>You must use the -f (force) option to drop an sbspace that contains data. <em>Restriction:</em> Use the -f option with sbspaces only. <strong>Warning:</strong> If you use the -f option, the tables in the database server might have dead pointers to the smart large objects that were deleted with this option. For more information, see dropping a chunk from an sbspace with onspaces, in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide.</em></td>
</tr>
<tr>
<td>blobspace</td>
<td>Names the blobspace to be dropped</td>
<td>Before you drop a blobspace, drop all tables that include a TEXT or BYTE column that references the blobspace.</td>
</tr>
<tr>
<td>dbspace</td>
<td>Names the dbspace to be dropped</td>
<td>Before you drop a dbspace, drop all databases and tables that you previously created in the dbspace.</td>
</tr>
<tr>
<td>extspace</td>
<td>Names the extspace to be dropped</td>
<td>You cannot drop an extspace if it is associated with an existing table or index.</td>
</tr>
<tr>
<td>sbspace</td>
<td>Names the sbspace to be dropped</td>
<td>Before you drop an sbspace, drop all tables that include a BLOB or CLOB column that references the sbspace.</td>
</tr>
</tbody>
</table>

**Important:** Do not specify a pathname when you drop these storage spaces.

### onspaces -f: Specify DATASKIP parameter

**Syntax:**

```plaintext
onspaces -f OFF ON dbspace-list -y
```

Use onspaces -f to specify the value of the DATASKIP configuration parameter. Use SET DATASKIP ON and SET DATASKIP OFF as the Administrative API command strings onspaces -f.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-f</td>
<td>Indicates to the database server that you want to change the DATASKIP default for specified dbspaces or all dbspaces</td>
<td>All changes in the DATASKIP status are recorded in the message log.</td>
</tr>
<tr>
<td>-y</td>
<td>Causes the database server to automatically respond yes to all prompts</td>
<td>None.</td>
</tr>
<tr>
<td>dbspace-list</td>
<td>Specifies the name of one or more dbspaces for which DATASKIP will be turned ON or OFF</td>
<td>Syntax must conform to the Identifier segment; see the <em>IBM Informix Guide to SQL: Syntax.</em> For more information, see “DATASKIP” on page 1-26 and the <em>IBM Informix Performance Guide.</em></td>
</tr>
<tr>
<td>OFF</td>
<td>Turns off DATASKIP</td>
<td>If you use OFF without dbspace-list, DATASKIP is turned off for all fragments. If you use OFF with dbspace-list, only the specified fragments are set with DATASKIP off.</td>
</tr>
<tr>
<td>ON</td>
<td>Turns on DATASKIP</td>
<td>If you use ON without dbspace-list, DATASKIP is turned on for all fragments. If you use ON with dbspace-list, only the specified fragments are set with DATASKIP on.</td>
</tr>
</tbody>
</table>
The `onspaces` utility lets you specify DATASKIP on a dbspace level or across all dbspaces.

### onspaces -m: Start mirroring

**Syntax:**

```
-mm dbspaces
```

Use `onspaces -m` to start mirroring for a dbspace, blobspace, or sbspace. Use `START MIRRORING` as the Administrative API `command` string for `onspaces -m`.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-f filename</code></td>
<td>Indicates that chunk-location information is in a file named <code>filename</code></td>
<td>The file must be a buffered file that already exists. The <code>filename</code> must conform to the operating-system-specific rules for pathnames. For more information, see &quot;Using a File to Specify Chunk-Location Information with the <code>-f</code> Option&quot; on page 14-23</td>
</tr>
<tr>
<td><code>-m</code></td>
<td>Adds mirroring for an existing dbspace, blobspace, or sbspace</td>
<td>User-data chunks in a mirrored sbspace need not be mirrored. The mirrored chunks should be on a different disk. You must mirror all the chunks at the same time.</td>
</tr>
<tr>
<td><code>-m pathname offset</code></td>
<td>The second time that <code>pathname</code> occurs in the syntax diagram, it indicates the disk partition or unbuffered device of the initial chunk of the dbspace, blobspace, or sbspace that performs the mirroring. The second time <code>offset</code> appears in the syntax diagram, it indicates the offset to reach the mirrored chunk of the newly mirrored dbspace, blobspace, or sbspace. Also see the entries for <code>pathname</code> and <code>offset</code> in this table.</td>
<td>None.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>-o offset</code></td>
<td>The first time that <code>offset</code> occurs in the syntax diagram, it indicates, in kilobytes, the offset into the disk partition or into the unbuffered device to reach the initial chunk of the newly mirrored dbspace, blobspace, or sbspace.</td>
<td><strong>Restrictions:</strong> Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes. For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td><code>-p pathname</code></td>
<td>The first time <code>pathname</code> occurs in the syntax diagram, it indicates the disk partition or unbuffered device of the initial chunk of the dbspace, blobspace, or sbspace that you want to mirror.</td>
<td>The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. For pathname syntax, see your operating-system documentation.</td>
</tr>
<tr>
<td><code>-y</code></td>
<td>Causes the database server to automatically respond yes to all prompts</td>
<td>None.</td>
</tr>
<tr>
<td><code>blobspace</code></td>
<td>Names the blobspace that you want to mirror</td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax. For more information, see the chapter on using mirroring in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td><code>dbspace</code></td>
<td>Names the dbspace that you want to mirror</td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax. For background information, see the chapter on using mirroring in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td><code>sbspace</code></td>
<td>Names the sbspace that you want to mirror</td>
<td>Syntax must conform to the Identifier segment; see IBM Informix Guide to SQL: Syntax. For background information, see the chapter on using mirroring in the IBM Informix Administrator’s Guide.</td>
</tr>
</tbody>
</table>

### Using a File to Specify Chunk-Location Information with the `-f` Option

You can create a file that contains the chunk-location information. Then, when you execute `onspaces`, use the `-f` option to indicate to the database server that this information is in a file whose name you specify in `filename`.

The contents of the file should conform to the following format, with options separated by spaces and each set of primary and mirror chunks on separate lines:

```
primary_chunk_path offset mirror_chunk_path offset
```

If the dbspace that you are mirroring contains multiple chunks, you must specify a mirror chunk for each of the primary chunks in the dbspace that you want to mirror. For an example that enables mirroring for a multichunk dbspace, see starting mirroring for unmirrored dbspaces with `onspaces` in the chapter on using mirroring in the IBM Informix Administrator’s Guide.
### onspaces -r: Stop mirroring

**Syntax:**

```
>> onspaces -r dbspace -y
```

Use `onspaces -r` to end mirroring for a dbspace, blobspace, or sbspace. Use STOP MIRRORING as the Administrative API command string for `onspaces -r`.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-r</td>
<td>Indicates to the database server that mirroring should be ended for an existing dbspace, blobspace, or sbspace.</td>
<td>For background information, see the chapter on using mirroring in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td>-y</td>
<td>Causes the database server to respond yes to all prompts automatically.</td>
<td>None.</td>
</tr>
<tr>
<td>blobspace</td>
<td>Names the blobspace for which you want to end mirroring.</td>
<td>Syntax must conform to the Identifier segment; see the <em>IBM Informix Guide to SQL: Syntax</em>. For more information, see the chapter on using mirroring in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td>dbspace</td>
<td>Names the dbspace for which you want to end mirroring.</td>
<td>Syntax must conform to the Identifier segment; see the <em>IBM Informix Guide to SQL: Syntax</em>. For more information, see the chapter on using mirroring in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td>sbspace</td>
<td>Names the sbspace for which you want to end mirroring.</td>
<td>Syntax must conform to the Identifier segment; see the <em>IBM Informix Guide to SQL: Syntax</em>. For background information, see the chapter on using mirroring in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
</tbody>
</table>

### onspaces -ren: Rename a dbspace, blobspace, sbspace, or extspace

**Syntax:**

```
>> onspaces -ren dbspace -n name
```

Use `onspaces -ren` to rename a dbspace, blobspace, sbspace, or extspace. Use RENAME SPACE as the Administrative API command string for `onspaces -ren`.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ren</td>
<td>Causes the database server to rename the specified blobspace, dbspace, extspace, or sbspace.</td>
<td><strong>Restrictions:</strong> You can rename a blobspace, dbspace, extspace, or sbspace when the database server is in quiescent mode. For more information, see the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-n name</td>
<td>Specifies the new name for the blobspace, dbspace, extspace, or sbspace</td>
<td><strong>Restrictions:</strong> The blobspace, dbspace, external space, or sbspace name must be unique and cannot exceed 128 bytes. It must begin with a letter or underscore and must contain only letters, numbers, underscores, or the $ character. For more information, see the chapter on managing disk space in the IBM Informix Administrator’s Guide. The syntax must conform to the Identifier segment. For more information, see the IBM Informix Guide to SQL: Syntax.</td>
</tr>
<tr>
<td>blobspace</td>
<td>Names the blobspace to be renamed</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For more information, see renaming spaces, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>dbspace</td>
<td>Names the dbspace to be renamed</td>
<td><strong>Restrictions:</strong> You cannot rename a critical dbspace, such as the root dbspace or a dbspace that contains physical logs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Additional Information:</strong> If you rename dbspaces that are included in the DATASKIP list, update the DATASKIP configuration parameter with the new names using the onspaces -f command. Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For more information, see renaming spaces, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>extspace</td>
<td>Names the extspace to be renamed</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For more information, see renaming spaces, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>sbspace</td>
<td>Names the sbspace to be renamed</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For more information, see renaming spaces, in the chapter on managing disk space in the IBM Informix Administrator’s Guide.</td>
</tr>
</tbody>
</table>

**Renaming a Dbspace, Blobspace, Sbspace, or Extspace when Enterprise Replication Is Active**

You can rename a space (dbspace, blobspace, sbspace, or extspace) when Enterprise Replication is active. When you put the database server into quiescent mode to rename the space, Enterprise Replication will be disconnected. You can then rename the space. The servers will resynchronize after you put the database server into online mode. If you want to rename the same space on another server, you must put that server into quiescent mode and rename the space separately. No enforced relationship is propagated between renamed spaces on different ER servers; the same tables can be in different spaces.

If the Enterprise Replication server also participates in High-Availability Data Replication (HDR), you can rename the dbspace on the primary server and it will be automatically propagate to the secondary server. (The secondary server cannot participate in Enterprise Replication.)

**Performing an Archive after Renaming a Space**

After renaming any space (except extspaces or temporary spaces), perform a level-0 archive of the renamed space and the root dbspace. This will ensure that
you can restore the spaces to a state including or following the rename dbspace operation. It is also necessary prior to performing any other type of archive.

**onspaces -s: Change status of a mirrored chunk**

**Syntax:**

```
  onspaces -s -D -o offset -O -p pathname -s blobspace -s sbspace
```

Use `onspaces -s` to change the status of a mirrored chunk in a dbspace, a non-primary chunk within a noncritical dbspace, a blobspace or an sbspace. Use `SET CHUNK ONLINE` or `SET CHUNK OFFLINE` as the Administrative API command string for `onspaces -s`.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-D</td>
<td>Indicates that you want to take the chunk down</td>
<td>None.</td>
</tr>
<tr>
<td>-o offset</td>
<td>Indicates, in kilobytes, the offset into the disk partition or unbuffered device to reach the chunk</td>
<td><strong>Restrictions:</strong> Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The offset must be a multiple of the page size. The maximum offset is 4 terabytes. For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td>-O</td>
<td>Indicates that you want to restore the chunk and bring it online</td>
<td>None.</td>
</tr>
<tr>
<td>-p pathname</td>
<td>Indicates the disk partition or unbuffered device of the chunk</td>
<td>The chunk can be an unbuffered device or a buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. For path name syntax, see your operating-system documentation.</td>
</tr>
<tr>
<td>-s</td>
<td>Indicates that you want to change the status of a chunk</td>
<td><strong>Restrictions:</strong> You can only change the status of a chunk in a mirrored pair or a non-primary chunk within a noncritical dbspace. For more information, see changing the mirror status in the <em>IBM Informix Administrator’s Guide</em>.</td>
</tr>
<tr>
<td>-y</td>
<td>Causes the database server to respond yes to all prompts automatically</td>
<td>None.</td>
</tr>
</tbody>
</table>

`blobspace` Names the blobspace whose status you want to change

Syntax must conform to the Identifier segment; see the *IBM Informix Guide to SQL: Syntax*. For more information, see changing the mirror status in the *IBM Informix Administrator’s Guide*. |
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbspace</td>
<td>Names the dbspace whose status you want to change</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For more information, see changing the mirror status in the IBM Informix Administrator’s Guide.</td>
</tr>
<tr>
<td>sb space</td>
<td>Names the sb space whose status you want to change</td>
<td>Syntax must conform to the Identifier segment; see the IBM Informix Guide to SQL: Syntax. For background information, see changing the mirror status in the IBM Informix Administrator’s Guide.</td>
</tr>
</tbody>
</table>
Chapter 15. The onstat Utility

In This Chapter .................................................. 15-3
Monitor the Database Server Status ............................................ 15-4
onstat Syntax ........................................................................ 15-4
   Statistics Collected from Source File ...................................... 15-7
   Interactive Execution ......................................................... 15-8
   Continuous onstat Execution ................................................ 15-8
onstat: onstat -pu equivalent .................................................... 15-8
onstat -p: Print output header .................................................. 15-8
   Logs Full Subheader ........................................................... 15-9
onstat --: Print onstat options and functions ............................... 15-9
onstat --option --infile: Print shared memory dump file ............... 15-9
onstat -a: Print onstat -cubtulp ............................................ 15-10
onstat -b: Print buffer information .......................................... 15-10
onstat -B: Print all buffer information ...................................... 15-12
onstat --: Print ONSCONFIG file contents ................................. 15-13
onstat -C: Print B-tree scanner information ............................... 15-13
onstat -d: Print chunk information .......................................... 15-14
   Using onstat -d with Sbspaces ......................................... 15-17
   Using onstat -d with Blobspaces ...................................... 15-17
onstat -D: Print page-read and page-write information .................. 15-18
onstat -f: Print dbspace information affected by datakip ............... 15-18
onstat -F: Print counts ....................................................... 15-19
onstat -g Monitoring Options ................................................ 15-20
   onstat -g act: Print active threads ...................................... 15-25
   onstat -g afr: Print allocated memory fragments ........................ 15-26
   onstat -g all: Print output from all onstat -g options ................. 15-26
   onstat -g ath: Print information about all threads ................. 15-26
   onstat -g buf: Print buffer pool profile information ................ 15-27
   onstat -g cat: Print ER global catalog information .................. 15-29
   onstat -g cdr config: Print ER settings ................................ 15-30
   onstat -g con: Print condition and thread information .............. 15-32
   onstat -g ddr: Print ER database log reader status ................. 15-33
   onstat -g dic: Print table information .................................. 15-34
   onstat -g dis: Print database server information ..................... 15-35
   onstat -g dll: Print dynamic libraries list ............................ 15-36
   onstat -g dmp: Print raw memory ........................................ 15-37
   onstat -g dri: Print HDR information ................................... 15-38
   onstat -g dsc: Print distribution cache information .................. 15-38
   onstat -g dss: Print ER environment data ............................... 15-39
   onstat -g dtc: Print delete table cleaner statistics .................. 15-40
   onstat -g env: Prints environment variable values ................... 15-41
   onstat -g ffr: Print free fragments ..................................... 15-43
   onstat -g glo: Print global multithreading information .............. 15-44
   onstat -g grp: Print ER grouper statistics ............................. 15-45
   onstat -g his: Print SQLTRACE information ............................ 15-50
   onstat -g ioa: Print combined onstat -g information ............... 15-52
   onstat -g iob: Print big buffer use summary ........................... 15-53
   onstat -g iof: Print asynchronous I/O statistics ..................... 15-54
   onstat -g iog: Print AIO global information ........................... 15-55
   onstat -g ioc: Print I/O queue information ............................ 15-55
   onstat -g ipl: Print index page logging status information ............ 15-57
   onstat -g iov: Print AIO VP statistics ................................ 15-57
   onstat -g iup: Print light append status information .................. 15-58
   onstat -g lm: Print all locked mutexes .................................. 15-59
   onstat -g lsc: Print active light scan status ........................... 15-60
onstat -g mem: Print pool memory statistics ........................................... 15-60
onstat -g mgm: Print MGM resource information ........................................ 15-61
onstat -g nbm: Print a block bit map ......................................................... 15-64
onstat -g nif: Print ER network interface statistics .................................... 15-65
onstat -g nsc client_id: Print current shared memory connection information .... 15-66
onstat -g nsl: Print poll threads shared-memory data .................................. 15-68
onstat -g nss: Print shared memory network connections status .................. 15-69
onstat -g ntd: Print network statistics ...................................................... 15-70
onstat -g ntm: Print network mail statistics .............................................. 15-70
onstat -g ntt: Print network user times ..................................................... 15-71
onstat -g ntu: Print network user statistics .............................................. 15-71
onstat -g opr: Print open partitions ............................................................ 15-72
onstat -g pos: Print file values ................................................................. 15-74
onstat -g ppp partition number | 0: Print partition profiles. ......................... 15-75
onstat -g prc: Print sessions using UDR or SPL routine ............................. 15-76
onstat -g queue: Prints ER queue statistics .............................................. 15-76
onstat -g qst: Print wait options for mutex and condition queues ................. 15-77
onstat -g rbf: Print a block map of shared memory .................................... 15-78
onstat -g rcv: Print ER receive manager statistics ...................................... 15-79
onstat -g rea: Print ready threads ............................................................. 15-82
onstat -g rep: Print ER schedule manager events ....................................... 15-82
onstat -g rqm: Print low-level queue statistics ......................................... 15-82
onstat -g rsv: Print RS secondary server information .................................. 15-85
onstat -g rwm: Print read and write mutexes ............................................. 15-88
onstat -g sch: Print VP information .......................................................... 15-88
onstat -g sds: Print SDS secondary server information ............................... 15-89
onstat -g seg: Print shared-memory segment statistics .................................. 15-93
onstat -g ses: Print session-related information ........................................ 15-93
onstat -g sle: Print all sleeping threads .................................................... 15-98
onstat -g smb: Print sbspaces information ................................................ 15-98
onstat -g smx: Print multiplexer group information .................................... 15-99
onstat -g spi: Print spin locks with long spins ......................................... 15-101
onstat -g sql: Print SQL-related session information ................................... 15-102
onstat -g src: Patterns in shared memory ................................................... 15-103
onstat -g ssc: Print SQL statement occurrences ....................................... 15-104
onstat -g stk tid: Print thread stack ......................................................... 15-106
onstat -g stm: Print SQL statement memory usage ...................................... 15-106
onstat -g stq: Print queue information ..................................................... 15-107
onstat -g sts: Print stack usage per thread ............................................... 15-107
onstat -g sync: Print ER synchronization status ........................................ 15-108
onstat -g tpf tid: Print thread profiles ..................................................... 15-110
onstat -g ufr: Print memory pool fragments ............................................. 15-110
onstat -g vcache: Print CPU VP memory block cache statistics .................... 15-112
onstat -g wait: Print wait queue thread list ............................................ 15-113
onstat -g wmx: Print all mutexes with waiters ........................................ 15-114
onstat -g wst: Print wait statistics for threads ......................................... 15-114
onstat -G: Print TP/XA transaction information ......................................... 15-116
onstat -h: Print buffer header hash chain information ................................. 15-117
onstat -i: Initiate interactive mode ............................................................ 15-118
onstat -j: Provide onplod load status information ..................................... 15-119
onstat -k: Print active lock information ..................................................... 15-121
onstat -l: Print physical and logical log information .................................... 15-122
onstat -m: Print recent system message log information ................................ 15-125
onstat -o: Output shared memory contents ............................................... 15-125
onstat -O: Print optical subsystem information ....................................... 15-126
onstat -p: Print profile counts ................................................................. 15-127
onstat -P: Print partition information ....................................................... 15-130
onstat -r: Repeatedly print selected statistics .......................................... 15-131
onstat -R: Print LRU, FRLU, and MRLU queue information ......................... 15-134
onstat -s: Print latch information ............................................................ 15-136
onstat -t and -T: Print tbspace information ............................................. 15-138
onstat -u: Print user activity profile ......................... 15-139
onstat -x: Print database server transaction information .................................................. 15-142
Determining the Position of a Logical-Log Record .............................................................. 15-143
Determining the Mode of a Global Transaction ..................................................................... 15-144
onstat -X: Print thread information ....................................................................................... 15-144
onstat -z: Clear statistics ...................................................................................................... 15-146
Return Codes on Exit. ........................................................................................................ 15-146

In This Chapter

This chapter shows you how to use the following onstat options:

- "onstat -:- Print output header” on page 15-8
- "onstat --: Print onstat options and functions” on page 15-9
- "onstat <option> <infile>: Print shared memory dump file” on page 15-9
- "onstat -a: Print onstat -cuskbdlp” on page 15-10
- "onstat -b: Print buffer information” on page 15-10
- "onstat -B: Print all buffer information” on page 15-12
- "onstat -c: Print ONCONFIG file contents” on page 15-13
- "onstat -C: Print B-tree scanner information” on page 15-13
- "onstat -d: Print chunk information” on page 15-14
- "onstat -D: Print page-read and page-write information” on page 15-18
- "onstat -F: Print counts” on page 15-19
- "onstat -g Monitoring Options” on page 15-20
- "onstat -G: Print TP/XA transaction information” on page 15-116
- "onstat -h: Print buffer header hash chain information” on page 15-117
- "onstat -j: Provide onUPLOAD status information” on page 15-119
- "onstat -k: Print active lock information” on page 15-121
- "onstat -l: Print physical and logical log information” on page 15-122
- "onstat -o: Output shared memory contents” on page 15-125
- "onstat -p: Print profile counts” on page 15-127
- "onstat -P: Print partition information” on page 15-130
- "onstat -r: Repeatedly print selected statistics” on page 15-131
- "onstat -t and -T: Print tblspace information” on page 15-138
- "onstat -u: Print user activity profile” on page 15-139
- "onstat -x: Print database server transaction information” on page 15-142
- "onstat -X: Print thread information” on page 15-144
- "onstat -z: Clear statistics” on page 15-146

The onstat utility reads shared-memory structures and provides statistics about the database server at the time that the command executes. The system-monitoring interface also provides information about the database server. For information on the system-monitoring interface, see Chapter 2, “The sysmaster Database.”

You can combine multiple onstat option flags in a single command. The contents of shared memory might change as the onstat output displays. The onstat utility does not place any locks on shared memory, so running the utility does not affect performance.
Monitor the Database Server Status

One useful feature of onstat output is the heading that indicates the database server status. Whenever the database server is blocked, onstat displays the following line after the banner line:

Blocked: reason

The variable reason can take one of the following values.

<table>
<thead>
<tr>
<th>Reason</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKPT</td>
<td>Checkpoint</td>
</tr>
<tr>
<td>LONGTX</td>
<td>Long transaction</td>
</tr>
<tr>
<td>ARCHIVE</td>
<td>Ongoing archive</td>
</tr>
<tr>
<td>MEDIA_FAILURE</td>
<td>Media failure</td>
</tr>
<tr>
<td>HANG_SYSTEM</td>
<td>Database server failure</td>
</tr>
<tr>
<td>DBS_DROP</td>
<td>Dropping a dbspaces</td>
</tr>
<tr>
<td>DDR</td>
<td>Discrete high-availability data replication</td>
</tr>
<tr>
<td>LBU</td>
<td>Logs full high-watermark</td>
</tr>
<tr>
<td>ADMINISTRATION</td>
<td>Database is in administration mode</td>
</tr>
</tbody>
</table>

onstat Syntax
Notes:
1 Only one occurrence of each item is allowed. More than one option can be specified on a single onstat command invocation.

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>Displays the output header</td>
<td>See &quot;onstat -a: Print output header&quot; on page 15-8</td>
</tr>
<tr>
<td>--</td>
<td>Displays a listing of all onstat options and their functions</td>
<td>This option is the only option flag that you cannot combine with any other flag. See &quot;onstat --: Print onstat options and functions&quot; on page 15-9</td>
</tr>
<tr>
<td>&lt;/option&gt; &lt;infile&gt;</td>
<td>Runs onstat commands against a shared memory dump file.</td>
<td>See &quot;onstat &lt;/option&gt; &lt;infile&gt;: Print shared memory dump file&quot; on page 15-9</td>
</tr>
<tr>
<td>-a</td>
<td>Interpreted as onstat -cuskbdlp. Displays output in that order.</td>
<td>See &quot;onstat -a: Print onstat -cuskbdlp&quot; on page 15-10</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Key Considerations</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>--------------------</td>
</tr>
<tr>
<td>-b</td>
<td>Displays information about buffers currently in use, including number of resident pages in the buffer pool</td>
<td>See “onstat -b: Print buffer information” on page 15-10</td>
</tr>
<tr>
<td>-B</td>
<td>Obtains information about all database server buffers, not just buffers currently in use.</td>
<td>See “onstat -B: Print all buffer information” on page 15-12</td>
</tr>
</tbody>
</table>
| -c      | Displays the ONCONFIG file:  
• $INFORMIXDIR/etc/ONCONFIG for UNIX  
• %INFORMIXDIR%/etc/%ONCONFIG% for Windows | See “onstat -c: Print ONCONFIG file contents” on page 15-13 |
<p>| -C      | Prints B-tree scanner information | See “onstat -C: Print B-tree scanner information” on page 15-13 |
| -d      | Displays information for chunks in each storage space | See “onstat -d: Print chunk information” on page 15-14 |
| -D      | Displays page-read and page-write information for the first 50 chunks in each dbspace | See “onstat -D: Print page-read and page-write information” on page 15-18 |
| -f      | Lists the dbspaces currently affected by the DATASKIP feature | See “onstat -f: Print dbspace information affected by dataskip” on page 15-18 |
| -F      | Displays a count for each type of write that flushes pages to disk | See “onstat -F: Print counts” on page 15-19 |
| -g      | Provides monitoring options | See “onstat -g Monitoring Options” on page 15-20 |
| -G      | Prints global transaction IDs | See “onstat -G: Print TP/XA transaction information” on page 15-116 |
| -h      | Provides information on the buffer header hash chains | See “onstat -h: Print buffer header hash chain information” on page 15-117 |
| -i      | Puts the onstat utility into interactive mode | See “onstat -i: Initiate interactive mode” on page 15-118 |
| -j      | Prints the interactive status of the active onupload process | See “onstat -j: Provide onupload status information” on page 15-119 |
| -k      | Displays information about active locks | See “onstat -k: Print active lock information” on page 15-121 |
| -l      | Displays information about physical and logical logs, including page addresses | See “onstat -l: Print physical and logical log information” on page 15-122 |
| -m      | Displays the 20 most recent lines of the database server message log | Output from this option lists the full pathname of the message-log file and the 20 file entries. A date-and-time header separates the entries for each day. A time stamp prefaces single entries within each day. The name of the message log is specified as MSGPATH in the ONCONFIG file. See “onstat -m: Print recent system message log information” on page 15-125 |
| -o      | Saves copies of the shared-memory segments to filename | See “onstat -o: Output shared memory contents” on page 15-125 |
| -O      | Displays information about the Optical Subsystem memory cache and staging-area blobspace | See “onstat -O: Print optical subsystem information” on page 15-126 |
| -p      | Displays profile counts. | See “onstat -p: Print profile counts” on page 15-127 |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-P</td>
<td>Displays for all partitions the partition number and the break-up of the buffer-pool pages that belong to the partition</td>
<td>See “onstat -P: Print partition information” on page 15-130</td>
</tr>
<tr>
<td>-r</td>
<td>Repeats the accompanying onstat options after they wait the specified seconds between each execution.</td>
<td>See “onstat -r: Repeatedly print selected statistics” on page 15-131</td>
</tr>
<tr>
<td>-R</td>
<td>Displays detailed information about the LRU queues, FLRU queues, and MLRU queues</td>
<td>See “onstat -R: Print LRU, FLRU, and MLRU queue information” on page 15-134</td>
</tr>
<tr>
<td>-s</td>
<td>Displays general latch information</td>
<td>See “onstat -s: Print latch information” on page 15-139</td>
</tr>
<tr>
<td>-t</td>
<td>Displays tblspace information, including residency state, for active tblspaces</td>
<td>See “onstat -t and -T: Print tblspace information” on page 15-138</td>
</tr>
<tr>
<td>-T</td>
<td>Displays tblspace information for all tblspaces</td>
<td>See “onstat -t and -T: Print tblspace information” on page 15-138</td>
</tr>
<tr>
<td>-u</td>
<td>Prints a profile of user activity</td>
<td>See “onstat -u: Print user activity profile” on page 15-139</td>
</tr>
<tr>
<td>-V</td>
<td>Displays the software version number and the serial number</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
<tr>
<td>-version</td>
<td>Displays the build version, host, OS, number and date, as well as the GLS version</td>
<td>See “Obtaining Utility Version Information” on page 6-2</td>
</tr>
<tr>
<td>-x</td>
<td>Displays information about transactions</td>
<td>See “onstat -x: Print database server transaction information” on page 15-142</td>
</tr>
<tr>
<td>-X</td>
<td>Obtains precise information about the threads that are sharing and waiting for buffers</td>
<td>See “onstat -X: Print thread information” on page 15-142</td>
</tr>
<tr>
<td>-z</td>
<td>Sets the profile counts to 0</td>
<td>See “onstat -z: Clear statistics” on page 15-146</td>
</tr>
<tr>
<td>filename_dest</td>
<td>Specifies destination file for the copy of the shared-memory segments Name must not match the name of any existing file. For pathname syntax, see your operating-system documentation.</td>
<td></td>
</tr>
<tr>
<td>filename_source</td>
<td>Specifies file that onstat reads as source for the requested information This file must include a previously stored shared-memory segment that you created with the -o option of onstat. For specific details on this option, see “Statistics Culled from Source File.” For pathname syntax, see your operating-system documentation.</td>
<td></td>
</tr>
<tr>
<td>Monitoring options</td>
<td>Specifies which onstat -g monitoring option to use</td>
<td>See “onstat -g Monitoring Options” on page 15-20</td>
</tr>
<tr>
<td>seconds</td>
<td>Specifies number of seconds between each execution of the onstat -r command This value must be an unsigned integer greater than 0.</td>
<td></td>
</tr>
</tbody>
</table>

## Statistics Culled from Source File

Use the `filename_source` parameter with other option flags to derive the requested onstat statistics from the shared-memory segments that `filename_source` contains. You must first use the onstat -o command to create a file that contains the shared-memory segments.
Interactive Execution

To put the onstat utility in interactive mode, use the -i option. Interactive mode allows you to enter multiple options, one after the other, without exiting the program. For information on using interactive mode, see “onstat -h: Print buffer header hash chain information” on page 15-117.

Continuous onstat Execution

Use the seconds parameter with the -r option flag to cause all other flags to execute repeatedly after they wait the specified seconds between each execution.

onstat: onstat -pu equivalent

Syntax:

```
--onstat
```

If you invoke onstat without any options, the command is interpreted as onstat -pu (-p option and -u option).

onstat -: Print output header

Syntax:

```
--onstat
```

All onstat output includes a header. The onstat - option displays only the output header and is useful for checking the database server mode. The header takes the following form:

```
Version--Mode (Type)--(Checkpnt)--Up Uptime--Sh_mem Kbytes
```

- **Version** is the product name and version number.
- **Mode** is the current operating mode.
- **(Type)** indicates whether the type is primary or secondary. If the database server is not involved in data replication, this field does not appear. If the type is primary, the value P appears. If the type is secondary, the value S appears.
- **(Checkpnt)** is a checkpoint flag. If it is set, the header might display two other fields after the mode if the timing is appropriate:
  - **(CKPT REQ)** indicates that a user thread has requested a checkpoint.
  - **(CKPT INP)** indicates that a checkpoint is in progress. During the checkpoint, access is limited to read only. The database server cannot write or update data until the checkpoint ends.
### Uptime
Indicates how long the database server has been running.

### \( Sh_{\text{mem}} \)
Is the size of database server shared memory, expressed in kilobytes.

A sample header for the database server follows:

## Logs Full Subheader
If the database server is blocked, the `onstat` header output includes an extra line that reads as follows:
**Blocked:** *reason(s)*

The reason can be one or more of the following.

<table>
<thead>
<tr>
<th>Reason</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKPT</td>
<td>Checkpoint</td>
</tr>
<tr>
<td>LONGTX</td>
<td>Long transaction</td>
</tr>
<tr>
<td>ARCHIVE</td>
<td>Ongoing storage-space backup</td>
</tr>
<tr>
<td>MEDIA_FAILURE</td>
<td>Media failure</td>
</tr>
<tr>
<td>HANG_SYSTEM</td>
<td>Database server failure</td>
</tr>
<tr>
<td>DBS_DROP</td>
<td>Dropping a dbspace</td>
</tr>
<tr>
<td>DDR</td>
<td>Discrete data replication</td>
</tr>
<tr>
<td>LBU</td>
<td>Logs full high-watermark</td>
</tr>
<tr>
<td>ADMINISTRATION</td>
<td>Database is in administration mode, with limited users, including user <em>informix</em></td>
</tr>
</tbody>
</table>

---

### onstat --: Print `onstat` options and functions

**Syntax:**

```
>>>onstat--
```

The `--` option displays a listing of all `onstat` options and their functions. This option is the only option flag that you cannot combine with any other flag.

### onstat `<option>` `<infile>`: Print shared memory dump file

**Syntax:**

```
>>>onstat- `<option>` `<infile>`
```

The `<option>` `<infile>` option enables you to run `onstat` commands against a shared memory dump file. The shared memory dump can be produced explicitly by using `onstat -o`. If the DUMPSHMEM configuration parameter is enabled on UNIX systems, the dump is created automatically at the time of an assertion failure. In addition to using `<option>` `<infile>` option, you can also use the `onstat -i` interactive mode.
When using the command line, the dump file must be entered as the final argument. The following example performs onstat -g ath against the shared memory dump contained in the file named dumpfile rather than attempting to attach to the shared memory of a running server.

```bash
onstat -g ath dumpfile
```

Using onstat -i interactive mode often saves time when more than one onstat command needs to be run against the dump file. When onstat -i is used, the file is read only once at the start; in command-line mode, each command reads the file.

The following example shows how to load a dump file in interactive mode:

```bash
onstat -i dumpfile
```

This reads in the shared memory dump and enters onstat interactive mode. Other onstat commands can be executed against the dump in the normal interactive fashion.

---

**onstat -a: Print onstat -cuskbtdlp**

**Syntax:**

```bash
onstat -a
```

The -a option is interpreted as onstat -cuskbtdlp, and output is displayed in that order. For an explanation of each option, refer to the appropriate flag in the paragraphs that follow.

---

**onstat -b: Print buffer information**

**Syntax:**

```bash
onstat -b
```

The onstat -b option displays information about buffers currently in use, including the total number of resident pages in the buffer pool. For information about displaying information about all buffers, use "onstat -B: Print all buffer information" on page 15-12.

The maximum number of buffers available is specified in the buffers field in the BUFFERPOOL configuration parameter in the ONCONFIG file.

The onstat -b option also provides summary information about the number of modified buffers, the total number of resident pages in the buffer pool, the total number of buffers available, the number of hash buckets available, and the size of the buffer in bytes (the page size).

```
123 modified, 23 resident, 2000 total, 2048 hash buckets, 2048 buffer size.
```

**Example Output:**

Buffer pool page size: 2048
4454970  0  84  1:30563  4472f000  18  801  80  ffffffffffffffff  0
4454d1b  0  84  1:30562  447b1800  18  801  80  ffffffffffffffff  45d654e0
4468b60  0  84  1:30567  4485e000  18  801  80  ffffffffffffffff  0
4476ec0  0  84  1:30565  44934000  18  801  80  ffffffffffffffff  0
44875b8  0  84  1:30564  44a2b800  18  801  80  ffffffffffffffff  0
449dc50  0  84  1:30563  44b7d000  18  801  80  ffffffffffffffff  0
44ad0700  0  c23  1:34245  44e78000  18  801  10  0  0
44ad1800  0  c23  1:34253  44e88000  18  801  10  0  0
44ad2900  0  c23  1:34261  44e98000  18  801  10  0  0
44ad3a00  0  c23  1:34269  44ea8000  18  801  10  0  0
44ad4b00  0  c23  1:34277  44eb8000  18  801  10  0  0
44ad5c00  0  c23  1:34285  44ec8000  18  801  10  0  0
44ad6d00  0  c23  1:34293  44ed7800  18  801  80  ffffffffffffffff  0
44af8e00  0  c23  1:34301  44ef8000  18  801  10  0  0
44a1a000  0  c23  1:34317  44f08000  18  801  10  0  0
44a1b100  0  c23  1:34325  44f18000  18  801  10  0  0
44a1c200  0  c23  1:34333  44f28000  18  801  10  0  0
44a1da80  0  c23  1:34351  44f38000  18  801  10  0  0
44a2d300  0  c23  1:34341  44f48000  18  801  10  0  0
44a3db80  0  c23  1:34346  44f58000  18  801  10  0  0
44a4c280  0  84  1:30569  4502b000  18  801  80  ffffffffffffffff  0
4472 modified, 5000 total, 8192 hash buckets, 2048 buffer size

Buffer pool page size: 8192
0 modified, 1000 total, 1024 hash buckets, 8192 buffer size

Figure 15-1. onstat -b Output

Output Description:

*Buffer pool page size* is the size of the buffer pool pages in bytes

*address* Is the address of the buffer header in the buffer table

*userthread* is the address of the most recent user thread to access the buffer table. Many user threads might be reading the same buffer concurrently.

*flgs* Uses the following flag bits to describe the buffer:

- 0x01 Modified data
- 0x02 Data
- 0x04 LRU
- 0x08 Error

*pagenum* Is the physical page number on the disk

*memaddr* Is the buffer memory address

*nslots* Is the number of slot-table entries in the page

This field indicates the number of rows (or portions of a row) that are stored on the page.

*pgflgs* Uses the following values, alone or in combination, to describe the page type:

- 1 Data page
2  Tblspace page
4  Free-list page
8  Chunk free-list page
9  Remainder data page
b  Partition resident blobpage
c  Blobspace resident blobpage
d  Blob chunk free-list bit page
e  Blob chunk blob map page
10  B-tree node page
20  B-tree root-node page
40  B-tree branch-node page
80  B-tree leaf-node page
100  Logical-log page
200  Last page of logical log
400  Sync page of logical log
800  Physical log
1000  Reserved root page
2000  No physical log required
8000  B-tree leaf with default flags

`xflags`  Uses the following flag bits to describe buffer access:
0x10  share lock
0x80  exclusive lock

`owner`  Is the user thread that set the `xflags` buffer flag
`waitlist`  Is the address of the first user thread that is waiting for access to this buffer

For a complete list of all threads waiting for the buffer, refer to
"onstat -X: Print thread information" on page 15-144

---

**onstat -B: Print all buffer information**

**Syntax:**

```
>> onstat -B
```

The `onstat -B` option displays information about all buffers. Both `onstat -B` and `onstat -b` display the same information except that `onstat -b` displays buffers that are currently being accessed while `onstat -B` displays information for all buffers. See "onstat -b: Print buffer information" on page 15-10 above for example output.
onstat -c: Print ONCONFIG file contents

Syntax:

```
onstat -c
```

Use the `onstat -c` option to display the contents of the ONCONFIG file. The database server first checks if you have assigned a value to the environment variable ONCONFIG. You can use the `onstat -c` option with the database server in any mode, including offline.

UNIX Only

On UNIX, if you have set ONCONFIG, `onstat -c` displays the contents of the $INFORMIXDIR/etc/ONCONFIG file. If not, by default, `onstat -c` displays the contents of $INFORMIXDIR/etc/onconfig.

End of UNIX Only

Windows Only

On Windows, if you have set ONCONFIG, `onstat -c` displays the contents of the %INFORMIXDIR%/etc\%ONCONFIG\% file. If not, by default, `onstat -c` displays the contents of %INFORMIXDIR%/etc\onconfig.

End of Windows Only

End of UNIX Only

onstat -C: Print B-tree scanner information

Syntax:

```
onstat -C prof
  hot
  part
  clean
  range
  map
  alice
  all
```

Use the `-C` option to print the information about the B-tree scanner subsystem and each B-tree scanner thread. The following options are available with the `onstat -C` command and can be combined:

- `prof` Prints the profile information for the system and each B-tree scanner thread. This is the default option.
- `hot` Prints the hot list index key in the order to be cleaned
- `part` Prints all partitions with index statistics
- `clean` Prints information about all the partitions that were cleaned or need to be cleaned
- `range` Prints the savings in pages processed by using index range scanning
map Displays the current bitmaps for each index being cleaned by the alice cleaning method
alice Displays the efficiency of the alice cleaning method option
all Prints all onstat -C options

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 03:59:17 -- 15360
Kbytes
Btree Cleaner Info
BT scanner profile Information
=======================================
Active Threads 1
Global Commands 20000 Building hot list
Number of partition scans 0
Main Block 0x0a69cc08
BTC Admin 0x0a4d9248

BTS info id Prio Partnum Key Cmd
0xa69cd5b 0 High 0x00000000 0 40 Yield N
  Number of leaves pages scanned 0
  Number of leaves with deleted items 0
  Time spent cleaning (sec) 0
  Number of index compresses 0
  Number of deleted items 0
  Number of index range scans 0
  Number of index leaf scans 0
```

Figure 15-2. onstat -C prof Output

**onstat -d: Print chunk information**

**Syntax:**

```
>>> onstat -d
```

Use the -d option to display information for chunks in each storage space. You can interpret output from this option as follows.

**Example Output:**
The first section of the display describes the storage spaces:

<table>
<thead>
<tr>
<th>address</th>
<th>flags</th>
<th>fchunk</th>
<th>nchunks</th>
<th>pgsizes</th>
<th>flags</th>
<th>owner</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>ad357e8</td>
<td>0x60001</td>
<td>1</td>
<td>1</td>
<td>2048</td>
<td>N B</td>
<td>informix</td>
<td>rootdbs</td>
</tr>
<tr>
<td>b62a5b0</td>
<td>0x60001</td>
<td>2</td>
<td>1</td>
<td>4096</td>
<td>N B</td>
<td>informix</td>
<td>dbsp1</td>
</tr>
</tbody>
</table>

2 active, 2047 maximum

NOTE: The values in the "page Rd" and "page Wr" columns for DBspace chunks are displayed in terms of system base page size.

Expanded chunk capacity mode: always

Output Description:

The first section of the display describes the storage spaces:

- **address**: Is the address of the storage space in the shared-memory space table.
- **number**: Is the unique ID number of the storage space assigned at creation.
- **flags**: Uses the following hexadecimal values to describe each storage space:
  - 0x00000000: Mirror not allowed and dbspace is unmirrored
  - 0x00000001: Mirror is allowed and dbspace is unmirrored
  - 0x00000002: Mirror is allowed and dbspace is mirrored
  - 0x00000004: Down
  - 0x00000008: Newly mirrored
  - 0x00000010: Blobspace
  - 0x00000020: Blobspace on removable media
  - 0x00000040: Blobspace is on optical media
  - 0x00000080: Blobspace is dropped
  - 0x00000100: Blobspace is the optical STAGEBLOB
  - 0x00000200: Space is being recovered
  - 0x00000400: Space is fully recovered
  - 0x00000800: Logical log is being recovered
  - 0x00001000: Table in dbspace is dropped
  - 0x00002000: Temporary dbspace
  - 0x00004000: Blobspace is being backed up
  - 0x00008000: Sbspace
  - 0x0000a001: Temporary sbspace
  - 0x00010000: Physical or logical log changed

Figure 15-3. onstat -d Output

Chapter 15. The onstat Utility
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00020000</td>
<td>Dbspace or chunk tables have changed</td>
</tr>
<tr>
<td>0x20002</td>
<td>Dbspace or chunk tables have changed and dbspace is mirrored</td>
</tr>
<tr>
<td>0x60001</td>
<td>Dbspace has large chunks and is unmirrored. Any changes result in changes on rootdbspace</td>
</tr>
</tbody>
</table>

- **fchunk**: Is the ID number of the first chunk
- **nchunks**: Is the number of chunks in the storage space
- **pgsize**: Is the size of the dbspace pages in bytes
- **flags**: Uses the following letter codes to describe each storage space:
  - **Position 1:**
    - M: Mirrored
    - N: Not mirrored
  - **Position 2:**
    - X: Newly mirrored
    - P: Physically recovered, waiting for P -- logical recovery
    - L: Being logically recovered
    - R: Being recovered
  - **Position 3:**
    - B: Blobspace
    - S: Sbspace
    - T: Temporary Dbspace
    - U: Temporary Sbspace
  - **Position 4:**
    - B: Dbspace has large chunks greater than 2 GB

- **owner**: Is the owner of the storage space
- **name**: Is the name of the storage space

In the line immediately following the storage-space list, **active** refers to the current number of storage spaces in the database server instance including the rootdbs and **maximum** refers to total **allowable** spaces for this database server instance.

The second section of the `onstat -d` output describes the chunks:

- **address**: Is the address of the chunk
- **chk/dbs**: Is the chunk number and the associated space number
- **offset**: Is the offset into the file or raw device in pages
- **size**: Is the size of the chunk in terms of the page size of the dbspace to which it belongs.
- **free**: Is the number of free pages in the chunk in terms of the page size of the dbspace to which it belongs.
For a blobspace, a tilde indicates an approximate number of free blobpages.

For an sbspace, indicates the number of free pages of user data space and total user data space.

*bpages*  
Is the size of the chunk in blobpages

Blobpages can be larger than disk pages; therefore, the *bpages* value can be less than the *size* value.

For an sbspace, is the size of the chunk in sbpages

*flags*  
Provides the chunk status information as follows:

**Position 1:**

P  Primary  
M  Mirror

**Position 2:**

N  Renamed and either Down or Inconsistent  
O  Online  
D  Down  
X  Newly mirrored  
I  Inconsistent

**Position 3:**

-  Dbspace  
B  Blobspace  
S  Sbspace

**Position 4:**

B  Has large chunks greater than 2 GB

*pathname*  
Is the pathname of the physical device

In the line immediately following the chunk list, **active** displays the number of active chunks (including the root chunk) and **maximum** displays the total number of chunks.

For information about page reads and page writes, refer to ["onstat -D: Print page-read and page-write information"](onstat-D: Print page-read and page-write information) on page 15-18.

**Using onstat -d with Sbspaces**

For information about using onstat -d to determine the size of sbspaces, user-data areas, and metadata areas, see monitoring sbspaces in the *IBM Informix Administrator's Guide.*

**Using onstat -d with Blobspaces**

If you issue the onstat -d command on an instance with blobspace chunks, the database server displays the following message:

**NOTE:** For BLOB chunks, the number of free pages shown is out of date.  
Run 'onstat -d update' for current stats.
To obtain the current statistics for blobspace chunks, issue the `onstat -d update` command. The `onstat` utility updates shared memory with an accurate count of free pages for each blobspace chunk. The database server displays the following message:

*Waiting for server to update BLOB chunk statistics...*

---

### onstat -D: Print page-read and page-write information

**Syntax:**

```
onstat -D
```

Use the `-D` option to display page-read and page-write information for the first 50 chunks in each space.

**Example Output:**

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 03:59:42 -- 34816 Kbytes

Dbspaces
address number flags fchunk nchunks pgsizes flags owner name
a40d7d8 1 0x1 1 1 2048 N informix rootdbs
1 active, 2047 maximum

Chunks
address chunk/dbs offset page Rd page Wr pathname
a40d928 1 1 0 0 0 /work/11.1/dbspaces/stardbs3
1 active, 2047 maximum

Expanded chunk capacity mode: disabled
```

*Figure 15-4. onstat -D Output*

**Output Description:**

The output of `onstat -D` is almost identical to the output of `onstat -d`. The following columns are unique to `onstat -D`. For information on the other output columns see "onstat -d: Print chunk information" on page 15-14.

- **Page Rd**
  - Is the number of pages read

- **Page Wr**
  - Is the number of pages written

### onstat -f: Print dbspace information affected by dataskip

**Syntax:**

```
onstat -f
```

Use the `-f` option to list the dbspaces that the dataskip feature currently affects. The `-f` option lists both the dbspaces that were set with the DATASKIP configuration parameter and the `-f` option of `onspaces`. When you execute `onstat -f`, the database server displays one of the following three outputs:

- **Dataskip is OFF for all dbspaces.**
• Dataskip is ON for all dbspaces.
• Dataskip is ON for the following dbspaces:
dbspace1 dbspace2...

onstat -F: Print counts

Syntax:

```
/onstat -F
```

Use the -F option to display a count for each type of write that flushes pages to disk.

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 04:00:17 -- 15360 Kbytes

Fg Writes  LRU Writes  Chunk Writes
0          0           0

address    flushe   state  data      = 0X0
a4d8628     0         1       0         = 0X0

states: Exit Idle Chunk Lru

Figure 15-5. onstat -F Output
```

Output Description:

You can interpret output from this option as follows:

- **Fg Writes**: Is the number of times that a foreground write occurred
- **LRU Writes**: Is the number of times that an LRU write occurred
- **Chunk Writes**: Is the number of times that a chunk write occurred
- **address**: Is the address of the user structure assigned to this page-cleaner thread
- **flushe**: Is the page-cleaner number
- **state**: Uses the following codes to indicate the current page-cleaner activity:
  - C  Chunk write
  - E  Exit
  - I  Cleaner is idle
  - L  LRU queue

The exit code indicates either that the database server is performing a shutdown or that a page cleaner did not return from its write in a specific amount of time. When an operation fails to complete within the allotted time, this situation is known as a
time-out condition. The database server does not know what happened to the cleaner, so it is marked as exit. In either case, the cleaner thread eventually exits.

*data* Provides additional information in concert with the *state* field

If *state* is C, *data* is the chunk number to which the page cleaner is writing buffers. If *state* is L, *data* is the LRU queue from which the page cleaner is writing. The *data* value is displayed as a decimal, followed by an equal sign, and repeated as a hexadecimal.

### **onstat -g Monitoring Options**

The following *onstat* -g options are provided for support and debugging only. You can include only one of these options per *onstat* -g command. For more information, see your IBM Informix Performance Guide.

<table>
<thead>
<tr>
<th><em>onstat</em> -g Option</th>
<th>Topic or Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>-g act</td>
<td>Prints active threads. See &quot;onstat -g act: Print active threads&quot; on page 15-25</td>
</tr>
<tr>
<td>-g aft pool name 1 session id</td>
<td>Prints allocated memory fragments for a specified session or shared-memory pool. Each session is allocated a pool of shared memory. To obtain the pool name, see the -mem option. See &quot;onstat -g aft: Print allocated memory fragments&quot; on page 15-26</td>
</tr>
<tr>
<td>-g all</td>
<td>Prints output from all <em>onstat</em> -g options. See &quot;onstat -g all: Print output from all <em>onstat</em> -g options&quot; on page 15-26</td>
</tr>
<tr>
<td>-g ath</td>
<td>Prints all threads. The sqlmain threads represent client sessions. The rsteb value corresponds to the user field of the <em>onstat</em> -u command. See &quot;onstat -g ath: Print information about all threads&quot; on page 15-26. For information on using <em>onstat</em> -g ath to print Enterprise Replication threads, see the IBM Informix Dynamic Server Enterprise Replication Guide.</td>
</tr>
<tr>
<td>-g buf</td>
<td>Print profile information for each buffer pool. See &quot;onstat -g buf: Print buffer pool profile information&quot; on page 15-27</td>
</tr>
<tr>
<td>-g cat [modifier]</td>
<td>Prints information from the Enterprise Replication global catalog. The global catalog contains a summary of information about the defined servers, replicates, and replicate sets on each of the servers within the enterprise. See &quot;onstat -g cat: Print ER global catalog information&quot; on page 15-29</td>
</tr>
<tr>
<td>-g cac agg</td>
<td>Prints the definitions for user-defined aggregates that are currently in the cache.</td>
</tr>
<tr>
<td>-g cac stmt</td>
<td>Prints the contents of the SQL statement cache. Prints the same output as the -g ssc statement.</td>
</tr>
<tr>
<td>-g con</td>
<td>Prints conditions with waiters. See &quot;onstat -g con: Print condition and thread information&quot; on page 15-32</td>
</tr>
<tr>
<td>-g cdr</td>
<td>Prints the settings of Enterprise Replication configuration parameters and environment variables. See &quot;onstat -g cdr config: Print ER settings&quot; on page 15-30</td>
</tr>
<tr>
<td>-g ddr</td>
<td>Prints the status of the Enterprise Replication database log reader. If log reading is blocked, data might not be replicated until the problem is resolved. See &quot;onstat -g ddr: Print ER database log reader status&quot; on page 15-33.</td>
</tr>
<tr>
<td><strong>onstat -g Option</strong></td>
<td><strong>Topic or Function</strong></td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td><strong>-g dic table</strong></td>
<td>Without any parameters, prints one line of information for each table cached in the shared-memory dictionary. If given a specific table name as a parameter, prints internal SQL information for that table. For more information, see your <em>IBM Informix Performance Guide</em>. For sample output, see “<em>onstat -g dic: Print table information</em>** on page 15-34</td>
</tr>
<tr>
<td><strong>-g dis</strong></td>
<td>Prints a list of database servers and their status, and information about each database server, <strong>INFORMIXDIR</strong>, <strong>sqlhosts</strong> file, <strong>ONCONFIG</strong> file, and hostname. See “<em>onstat -g dis: Print database server information</em>** on page 15-35</td>
</tr>
<tr>
<td><strong>-g dll</strong></td>
<td>Prints a list of dynamic libraries that have been loaded. See “<em>onstat -g dll: Print database server information</em>** on page 15-35</td>
</tr>
<tr>
<td><strong>-g dmp</strong></td>
<td>Prints prints raw memory at a given address for a number of given bytes. See “<em>onstat -g dmp: Print raw memory</em>** on page 15-37</td>
</tr>
<tr>
<td><strong>-g dri</strong></td>
<td>Prints data-replication information. See monitoring High-Availability Data-Replication status in the <em>IBM Informix Administrator’s Guide</em>. See “<em>onstat -g dri: Print HDR information</em>** on page 15-38</td>
</tr>
<tr>
<td><strong>-g dsc</strong></td>
<td>Prints data-distribution cache information. See “<em>onstat -g dsc: Print distribution cache information</em>** on page 15-38</td>
</tr>
<tr>
<td><strong>-g dss [modifier]</strong></td>
<td>Prints detailed statistical information about the activity of individual data sync threads and about user-defined data types. See “<em>onstat -g dss: Print ER environment data</em>** on page 15-39</td>
</tr>
<tr>
<td><strong>-g dtc</strong></td>
<td>Prints statistics about the delete table cleaner which removes rows from the delete table when they are no longer needed. See “<em>onstat -g dtc: Print delete table cleaner statistics</em>** on page 15-40</td>
</tr>
<tr>
<td><strong>-g env</strong></td>
<td>Prints the values of environment variables the database server currently uses. See “<em>onstat -g env: Prints environment variable values</em>** on page 15-41</td>
</tr>
<tr>
<td><strong>-g ffr pool name [session id]</strong></td>
<td>Prints free fragments for a pool of shared memory. See “<em>onstat -g ffr: Print free fragments</em>** on page 15-43</td>
</tr>
<tr>
<td><strong>-g glo</strong></td>
<td>Prints global multithreading information. This information includes CPU use information about the virtual processors, the total number of sessions, and other multithreading global counters. On Windows, the virtual processors are operating system threads. The values displayed under the ‘pid’ field are thread ids not process ids. (Windows). See “<em>onstat -g glo: Print global multithreading information</em>** on page 15-44</td>
</tr>
<tr>
<td><strong>-g grp [modifier]</strong></td>
<td>Prints statistics about the Enterprise Replication grouper. The grouper evaluates the log records, rebuilds the individual log records into the original transaction, packages the transaction, and queues the transaction for transmission. See “<em>onstat -g grp: Print ER grouper statistics</em>** on page 15-45</td>
</tr>
<tr>
<td><strong>-g his</strong></td>
<td>Prints information about the SQLTrace configuration parameter. See “<em>onstat -g his: Print SQLTRACE information</em>** on page 15-50</td>
</tr>
<tr>
<td><strong>-g imc</strong></td>
<td>Prints information about MaxConnect instances that are connected to the database server. If MaxConnect is not connected to the database server, this command displays No MaxConnect servers are connected.</td>
</tr>
<tr>
<td><strong>-g ioa</strong></td>
<td>Prints combined information from -g ioq and -g iov. See “<em>onstat -g ioa: Print combined onstat -g information</em>** on page 15-52</td>
</tr>
<tr>
<td>onstat -g Option</td>
<td>Topic or Function</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>-g iob</td>
<td>Prints the big buffer usage summary. See “onstat -g iob: Print big buffer use summary” on page 15-53</td>
</tr>
<tr>
<td>-g iof</td>
<td>Prints asynchronous I/O statistics by chunk or file. This option is similar to the -D option, except it also displays information on nonchunk, temporary, and sort-work files. See “onstat -g iof: Print asynchronous I/O statistics” on page 15-54</td>
</tr>
<tr>
<td>-g iog</td>
<td>Prints AIO global information. See “onstat -g iog: Print AIO global information” on page 15-55</td>
</tr>
<tr>
<td>-g ioq queue name</td>
<td>Prints pending I/O operations for the queue name. If given the gfd or kaito queue name, a queue for each CPU VP is displayed. If queue name is omitted, I/O statistics for all queues are displayed. See “onstat -g ioq: Print I/O queue information” on page 15-55</td>
</tr>
<tr>
<td>-g iov</td>
<td>Prints asynchronous I/O statistics by virtual processor. See “onstat -g iov: Print AIO VP statistics” on page 15-57</td>
</tr>
<tr>
<td>-g ipl</td>
<td>Prints index page logging status. See “onstat -g ipl: Print index page logging status information” on page 15-57</td>
</tr>
<tr>
<td>-g lap</td>
<td>Prints information on the status of light appends. See “onstat -g lap: Print light append status information” on page 15-58</td>
</tr>
<tr>
<td>-g lmx</td>
<td>Prints all locked mutexes. See “onstat -g lmx: Print all locked mutexes” on page 15-59</td>
</tr>
<tr>
<td>-g lsc</td>
<td>Displays information about light scans. See “onstat -g lsc: Print active light scan status” on page 15-60</td>
</tr>
<tr>
<td>-g mem pool name</td>
<td>Prints statistics for a memory pool. Also displays the pool name, type of shared memory segment that contains the pool, the address of the pool, the total size of the pool, the number of bytes of free memory that it contains, and the number of free and allocated fragments in the pool. If no argument is provided, displays information about all pools. The block pools are listed in a separate section after the main pool list. You also can use ISA to obtain detailed information about a memory pool. If you run an SQL query that allocates memory from the PER_STMT_EXEC and PER_STMT_PREP memory duration pools, onstat -g mem displays information on the PRPsessionid.threadid pool and the EXE.sessionid.threadid pool. See “onstat -g mem: Print pool memory statistics” on page 15-60. For more information, see the IBM Informix DataBlade API Programmer’s Guide.</td>
</tr>
<tr>
<td>-g mgm</td>
<td>Prints Memory Grant Manager resource information. See “onstat -g mgm: Print MGM resource information” on page 15-61</td>
</tr>
<tr>
<td>-g nbm</td>
<td>Prints block bit map for the nonresident segments, one bit per 8-kilobyte block. Bit set indicates block free. See “onstat -g nbm: Print a block bit map” on page 15-64</td>
</tr>
<tr>
<td>-g nif [modifier]</td>
<td>Prints statistics about the network interface. Useful to determine why data is not replicating. See “onstat -g nif: Print ER network interface statistics” on page 15-65</td>
</tr>
<tr>
<td>-g nsc client id</td>
<td>Prints shared-memory status by client id. If client id is omitted, all client status areas are displayed. This command prints the same status data as the nss command. See “onstat -g nsc client id: Print current shared memory connection information” on page 15-66</td>
</tr>
<tr>
<td>-g nsd</td>
<td>Prints network shared-memory data for poll threads. See “onstat -g nsd: Print poll threads shared-memory data” on page 15-68</td>
</tr>
<tr>
<td>onstat -g Option</td>
<td>Topic or Function</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><code>-g nss session id</code></td>
<td>Prints network shared-memory status by <code>session id</code>. If <code>session id</code> is omitted, all session status areas are displayed. This command prints the same status data as the <code>-nsc</code> command. See &quot;onstat -g nss: Print shared memory network connections status&quot; on page 15-69.</td>
</tr>
<tr>
<td><code>-g nta</code></td>
<td>Prints combined network statistics from <code>-g ntd</code>, <code>-g ntm</code>, <code>-g ntt</code>, and <code>-g ntu</code>. If MaxConnect is installed, this command prints statistics that you can use to tune MaxConnect performance.</td>
</tr>
<tr>
<td><code>-g ntd</code></td>
<td>Prints network statistics by service. See &quot;onstat -g ntd: Print network statistics&quot; on page 15-70.</td>
</tr>
<tr>
<td><code>-g ntm</code></td>
<td>Prints network mail statistics. See &quot;onstat -g ntm: Print network mail statistics&quot; on page 15-70.</td>
</tr>
<tr>
<td><code>-g ntt</code></td>
<td>Prints network user times. See &quot;onstat -g ntt: Print network user times&quot; on page 15-71.</td>
</tr>
<tr>
<td><code>-g ntu</code></td>
<td>Prints network user statistics. See &quot;onstat -g ntu: Print network user statistics&quot; on page 15-71.</td>
</tr>
<tr>
<td><code>-g opn</code></td>
<td>Prints open partitions. See &quot;onstat -g opn: Print open partitions&quot; on page 15-72.</td>
</tr>
<tr>
<td><code>-g pos</code></td>
<td>Prints <code>INFORMIXDIR/etc/infos=DBSERVERNAME</code> file for UNIX and <code>%INFORMIXDIR%/etc\infos=DBSERVERNAME</code> for Windows. See &quot;onstat -g pos: Print file values&quot; on page 15-74.</td>
</tr>
<tr>
<td><code>-g ppf partition number 0</code></td>
<td>Prints partition profile for <code>partition number</code>: 0 prints profiles for all partitions. If TBLSPACE_STATS configuration parameter is set to 0, displays: Partition profiles is disabled. See &quot;onstat -g ppf partition number 0: Print partition profiles&quot; on page 15-75.</td>
</tr>
<tr>
<td><code>-g prc</code></td>
<td>Prints information about SPL routine cache. See &quot;onstat -g prc: Print sessions using UDR or SPL routine&quot; on page 15-76.</td>
</tr>
<tr>
<td><code>-g qst</code></td>
<td>Prints queue wait statistics. See &quot;onstat -g qst: Print wait options for mutex and condition queues&quot; on page 15-77.</td>
</tr>
<tr>
<td><code>-g que</code></td>
<td>Prints statistics for the high-level queue interface (which are common to all the queues of the Enterprise Replication Queue Manager. See &quot;onstat -g que: Prints ER queue statistics&quot; on page 15-76.</td>
</tr>
<tr>
<td><code>-g rbm</code></td>
<td>Prints block bit map for the resident segment (communication message area). See &quot;onstat -g rbm: Print a block map of shared memory&quot; on page 15-78.</td>
</tr>
<tr>
<td><code>-g rcv [serverid]</code></td>
<td>Prints statistics about the receive manager, which is a set of service routines between the receive queues and data sync. See &quot;onstat -g rcv: Print ER receive manager statistics&quot; on page 15-79.</td>
</tr>
<tr>
<td><code>-g rea</code></td>
<td>Prints ready threads. See &quot;onstat -g rea: Print ready threads&quot; on page 15-82.</td>
</tr>
<tr>
<td><code>-g rep [replname]</code></td>
<td>Prints events that are in the queue for the schedule manager. See &quot;onstat -g rep: Print ER schedule manager events&quot; on page 15-82.</td>
</tr>
<tr>
<td><code>-g rqm [modifier]</code></td>
<td>Prints statistics and contents of the low-level queues (each individual queue) managed by the Reliable Queue Manager (RQM). See &quot;onstat -g rqm: Print low-level queue statistics&quot; on page 15-82.</td>
</tr>
<tr>
<td><code>-g rss [modifier]</code></td>
<td>Prints remote standalone secondary (RSS) server information. See &quot;onstat -g rss: Print RS secondary server information&quot; on page 15-83.</td>
</tr>
<tr>
<td><code>-g rwm</code></td>
<td>Prints read/write mutexes. See &quot;onstat -g rwm: Print read and write mutexes&quot; on page 15-88.</td>
</tr>
<tr>
<td>onstat -g Option</td>
<td>Topic or Function</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>-g sch</td>
<td>Prints the number of semaphore operations, spins, and busy waits for each virtual processor. On Windows, the virtual processors are operating system threads. The values displayed under the ‘pid’ field are thread ids not process ids. (Windows) See “onstat -g sch: Print Virtual information” on page 15-88</td>
</tr>
<tr>
<td>-g sds [ modifier]</td>
<td>Prints shared disk secondary (SDS) server information. See “onstat -g sds: Print SDS secondary server information” on page 15-89</td>
</tr>
<tr>
<td>-g seg</td>
<td>Prints shared-memory-segment statistics. This option shows the number and size of shared-memory segments that the database server is currently using. See “onstat -g seg: Print shared-memory segment statistics” on page 15-93</td>
</tr>
<tr>
<td>-g ses sessionid</td>
<td>Prints session information by sessionid. If sessionid is missing, a one-line summary of each session prints. See “onstat -g ses: Print session-related information” on page 15-93</td>
</tr>
<tr>
<td>-g sle</td>
<td>Prints all sleeping threads. See “onstat -g sle: Print all sleeping threads” on page 15-98</td>
</tr>
<tr>
<td>-g smb option</td>
<td>Prints detailed information about sbspaces. See “onstat -g smb: Print sbspaces information” on page 15-98</td>
</tr>
<tr>
<td>-g smx option</td>
<td>Displays server multiplexer group connections information. See “onstat -g smx: Print multiplexer group information” on page 15-99</td>
</tr>
<tr>
<td>-g spi</td>
<td>Prints spin locks with long spins. See “onstat -g spi: Print spin locks with long spins” on page 15-101</td>
</tr>
<tr>
<td>-g sql session id</td>
<td>Prints SQL information by session id. If session id is omitted, a one-line summary for each session prints. See “onstat -g sql: Print SQL-related session information” on page 15-102</td>
</tr>
<tr>
<td>-g src session id</td>
<td>Searches for patterns in shared memory. See “onstat -g src: Patterns in shared memory” on page 15-103</td>
</tr>
<tr>
<td>-g ssc</td>
<td>Monitors the number of times that the database server reads the SQL statement in the cache. See example output, see “onstat -g ssc: Print SQL statement occurrences” on page 15-104</td>
</tr>
<tr>
<td></td>
<td>Displays the same output as onstat -g cac stmt. For more information, see improving query performance in the IBM Informix Performance Guide.</td>
</tr>
<tr>
<td>-g ssc all</td>
<td>Reports the key-only cache entries as well as the fully cached statements. If the value in the hits column is less than the STMT_CACHE_HITS value, that entry is a key-only cache entry. For more information, see memory utilization in the IBM Informix Performance Guide.</td>
</tr>
<tr>
<td>-g ssc pool</td>
<td>Reports usage of all memory pools for the SQL statement cache. The output displays information on the name, class, address, and total size of the memory pools. For more information, see improving query performance in the IBM Informix Performance Guide.</td>
</tr>
<tr>
<td>-g stk tid</td>
<td>Dumps stack of thread specified by thread ID or stacks for all threads. This option is not supported on all platforms and is not always accurate. See “onstat -g stk tid: Print thread stack” on page 15-106</td>
</tr>
<tr>
<td>onstat -g Option</td>
<td>Topic or Function</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>-g stm [session id]</td>
<td>Displays the memory that each prepared SQL statement uses. See &quot;onstat -g stm: Print SQL statement memory usage&quot; on page 15-106. For more information, see memory utilization and improving query performance in the IBM Informix Performance Guide.</td>
</tr>
<tr>
<td>-g stq</td>
<td>Prints queue stream information. See &quot;onstat -g stq: Print queue information&quot; on page 15-107.</td>
</tr>
<tr>
<td>-g sts</td>
<td>Prints maximum and current stack use per thread. See &quot;onstat -g sts: Print stack usage per thread&quot; on page 15-107.</td>
</tr>
<tr>
<td>-g sync</td>
<td>Shows which sync is active. See &quot;onstat -g sync: Print ER synchronization status&quot; on page 15-108.</td>
</tr>
<tr>
<td>-g tpf tid</td>
<td>Prints thread profile for a specific thread ID. See &quot;onstat -g tpf tid: Print thread profiles&quot; on page 15-110.</td>
</tr>
<tr>
<td>-g ufr pool name</td>
<td>Prints allocated fragments by use. See &quot;onstat -g ufr: Print memory pool fragments&quot; on page 15-110.</td>
</tr>
<tr>
<td>-g ufr pool name</td>
<td>Prints wait statistics for threads. See &quot;onstat -g wst: Print wait statistics for threads&quot; on page 15-114.</td>
</tr>
<tr>
<td>-g vpcache</td>
<td>Returns information about CPU VP memory block cache statistics. See &quot;onstat -g vpcache: Print CPU VP memory block cache statistics&quot; on page 15-112.</td>
</tr>
<tr>
<td>-g wai</td>
<td>Prints waiting threads; all threads waiting on mutex or condition, or yielding. See &quot;onstat -g wai: Print wait queue thread list&quot; on page 15-113.</td>
</tr>
<tr>
<td>-g wmx</td>
<td>Prints all mutexes with waiters. See &quot;onstat -g wmx: Print all mutexes with waiters&quot; on page 15-114.</td>
</tr>
</tbody>
</table>

**onstat -g act: Print active threads**

**Syntax:**

```plaintext
onstat -g act
```

The `onstat -g act` option prints active threads.

Following is sample output from the `onstat -g act` command. For a description of the output, see "onstat -g ath: Print information about all threads" on page 15-26.

```
-- 101376 Kbytes
Running threads:
  tid  tcb  rstcb  prty  status  vp-class  #schilds name
  2  b3132d8  0  1  running  2adm  0  adminthd
  40  c5384d0  0  1  running  1cpu  102630  t1ftcppoll
```

*Figure 15-6. onstat -g act Output*
onstat -g afr: Print allocated memory fragments

Syntax:

```bash
onstat -g afr
```

The `onstat -g afr` option prints allocated memory fragments for a specified session or shared-memory pool. Each session is allocated a pool of shared memory.

Example Output:

```
-- 43008 Kbytes
Allocations for pool name dfm_pool:
addr  size  memid
10ac8c000  192  overhead
10ac8d000 24352  dfm
```

*Figure 15-7. onstat -g afr Output*

**Output Description:**

- `addr`: Memory address of the pool fragment
- `size`: Size, in bytes, of the pool fragment
- `memid`: Memory ID of the pool fragment

onstat -g all: Print output from all onstat -g options

Syntax:

```bash
onstat -g all
```

The `onstat -g all` option prints output from all of the `onstat -g` options and is primarily used by IBM Support to gather diagnostic information. For normal administrative purposes, only the individual options should be invoked.

onstat -g ath: Print information about all threads

Syntax:

```bash
onstat -g ath
```

The `onstat -g ath` option prints information about all threads.

Example Output:
**Output Description:**

- **tid**: Thread ID
- **tcb**: Thread control block access
- **rstcb**: RSAM thread control block access
- **prty**: Thread priority
- **status**: Thread status
- **vp-class**: Virtual processor class
- **name**: Thread name

**onstat -g buf: Print buffer pool profile information**

**Syntax:**

```
onstat -g buf```

The **onstat -g buf** option prints profile information for each buffer pool.

**Example Output:**

```
Figure 15-8. onstat -g ath Output

<table>
<thead>
<tr>
<th>tid</th>
<th>tcb</th>
<th>rstcb</th>
<th>prty</th>
<th>status</th>
<th>vp-class</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10bbf36a8</td>
<td>0</td>
<td>1</td>
<td>IO</td>
<td>3lio</td>
<td>lio vp 0</td>
</tr>
<tr>
<td>3</td>
<td>10bc12218</td>
<td>0</td>
<td>1</td>
<td>IO</td>
<td>4pio</td>
<td>pio vp 0</td>
</tr>
<tr>
<td>4</td>
<td>10bc31218</td>
<td>0</td>
<td>1</td>
<td>running</td>
<td>5aio</td>
<td>aio vp 0</td>
</tr>
<tr>
<td>5</td>
<td>10bc50218</td>
<td>0</td>
<td>1</td>
<td>IO</td>
<td>6msc</td>
<td>msc vp 0</td>
</tr>
<tr>
<td>6</td>
<td>10bc7f218</td>
<td>0</td>
<td>1</td>
<td>running</td>
<td>7aio</td>
<td>aio vp 1</td>
</tr>
<tr>
<td>7</td>
<td>10bc9e540</td>
<td>10b231028</td>
<td>1</td>
<td>sleeping secs: 1</td>
<td>1cpu</td>
<td>main_loop()</td>
</tr>
<tr>
<td>8</td>
<td>10bc2548</td>
<td>0</td>
<td>1</td>
<td>running</td>
<td>1cpu</td>
<td>tltcppoll</td>
</tr>
<tr>
<td>9</td>
<td>10bc317f0</td>
<td>0</td>
<td>1</td>
<td>sleeping forever</td>
<td>1cpu</td>
<td>tltcppoll</td>
</tr>
<tr>
<td>10</td>
<td>10bc50438</td>
<td>10b231780</td>
<td>1</td>
<td>IO Wait</td>
<td>1cpu</td>
<td>flush_sub(0)</td>
</tr>
<tr>
<td>11</td>
<td>10bc7f740</td>
<td>0</td>
<td>1</td>
<td>IO</td>
<td>8aio</td>
<td>aio vp 2</td>
</tr>
<tr>
<td>12</td>
<td>10bc7fa00</td>
<td>0</td>
<td>1</td>
<td>IO</td>
<td>9aio</td>
<td>aio vp 3</td>
</tr>
<tr>
<td>13</td>
<td>10bd56218</td>
<td>0</td>
<td>1</td>
<td>IO</td>
<td>10aio</td>
<td>aio vp 4</td>
</tr>
<tr>
<td>14</td>
<td>10bd75218</td>
<td>0</td>
<td>1</td>
<td>IO</td>
<td>11aio</td>
<td>aio vp 5</td>
</tr>
<tr>
<td>15</td>
<td>10bd94548</td>
<td>10b231ed8</td>
<td>1</td>
<td>sleeping forever</td>
<td>1cpu</td>
<td>aslogflush</td>
</tr>
<tr>
<td>16</td>
<td>10bc7fd00</td>
<td>10b232630</td>
<td>1</td>
<td>sleeping secs: 34</td>
<td>1cpu</td>
<td>btscanner 0</td>
</tr>
<tr>
<td>32</td>
<td>10c738ad8</td>
<td>10b233c38</td>
<td>1</td>
<td>sleeping secs: 1</td>
<td>1cpu</td>
<td>onmode_mon</td>
</tr>
<tr>
<td>50</td>
<td>10c0db710</td>
<td>10b232d88</td>
<td>1</td>
<td>IO Wait</td>
<td>1cpu</td>
<td>sqlexec</td>
</tr>
</tbody>
</table>
```

Chapter 15. The onstat Utility  15-27
IBM Informix Dynamic Server Version 11.10.F -- On-Line -- Up 00:00:25 -- 1075788 Kbytes

Profile

<table>
<thead>
<tr>
<th>Buffer pool page size: 2048</th>
<th>dskreads</th>
<th>pagreads</th>
<th>bufreads</th>
<th>%cached</th>
<th>dskwrits</th>
<th>pagwrits</th>
<th>bufwrits</th>
<th>%cached</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2065</td>
<td>2067</td>
<td>274619</td>
<td>99.25</td>
<td>4418</td>
<td>36043</td>
<td>81649</td>
<td>94.59</td>
</tr>
<tr>
<td>(bufwrites_sinceckpt)</td>
<td>14850</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>Bufwrits sinceckpt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bufwrits</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Bufwaits</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>%cached</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Avg.</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Avg. LRU Time</td>
<td>nan</td>
<td>nan</td>
<td>nan</td>
<td>nan</td>
<td>nan</td>
<td>nan</td>
<td>nan</td>
<td>nan</td>
</tr>
<tr>
<td>Avg. LRU Chunk Writes</td>
<td>2909</td>
<td>2909</td>
<td>2909</td>
<td>2909</td>
<td>2909</td>
<td>2909</td>
<td>2909</td>
<td>2909</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Buffer pool page size: 8192</th>
</tr>
</thead>
<tbody>
<tr>
<td>dskreads</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>bufwrites_sinceckpt</td>
</tr>
<tr>
<td>Bufwrits sinceckpt</td>
</tr>
<tr>
<td>Bufwrits</td>
</tr>
<tr>
<td>Bufwaits</td>
</tr>
<tr>
<td>%cached</td>
</tr>
<tr>
<td>Avg.</td>
</tr>
<tr>
<td>Avg. LRU Time</td>
</tr>
<tr>
<td>Avg. LRU Chunk Writes</td>
</tr>
</tbody>
</table>

**Figure 15-9. onstat -g buf Output**

**Output Description:**

*Buffer pool page size*

- **dskreads**: Number of disk read operations performed to bring pages into this buffer pool. Each read operation reads one or more pages.
- **pagreads**: Number of pages read from disk to this buffer pool.
- **bufreads**: Number of times a memory image for a page was read from this buffer pool.
- **%cached**: Percentage of page reads for this buffer pool that were satisfied by a cached page image (rather than having to perform a disk read). Computed as (bufreads - dskreads) / bufreads x 100. Higher percentages indicate better caching performance.
- **dskwrits**: Number of disk write operations performed to write changed pages from this buffer pool back to disk. Each write operation writes one or more pages.
- **pagwrits**: Number of pages written to disk from this buffer pool.
- **bufwrits**: Number of times a memory image of a page was written to in this buffer pool.
- **%cached**: Percentage of page writes for this buffer pool that were satisfied by a cached page image (rather than having to perform a disk write). Computed as (bufwrits - dskwrits) / bufwrits x 100.
- **bufwrites_sinceckpt**: Number of times a memory image of a page was written to in this buffer pool since the last checkpoint.
- **bufwaits**: Number of times a thread had to wait for a lock on a buffer in this buffer pool. Higher numbers indicate more contention among multiple threads for mutually incompatible locks on the same pages.
- **ovbuff**: Number of times a changed buffer from this buffer pool was written to disk specifically to create a free buffer to read another requested page. If the ovbuff value is high, it may indicate that the buffer pool is not large enough to hold the working set of pages.
needed by the applications using this buffer pool, which may lead to performance degradation from I/O thrashing.

**flushes**
Number of times the server performed a mass flush of all dirty buffers in the buffer pool. This can occur for a variety of reasons, including as part of checkpoint processing or if the buffer pool is running out of clean buffers despite normal LRU cleaning activity.

**Fg Writes**
Number of changed buffers from this buffer pool that were written to disk by a non-I/O flusher thread that was accessing the buffer. This number is a superset of ovbuff. In addition to the writes to service page faults counted by ovbuff, this value also includes foreground writes that are done by certain operations to maintain the consistency of database logs and reserved pages in order to guarantee correctness of crash recovery in special cases.

**LRU Writes**
Number of changed buffers from this buffer pool that were written to disk by an LRU cleaner thread. LRU cleaners are activated if the buffer pool exceeds its LRU_MAX_DIRTY threshold or if foreground writes occur due to buffer pool overflows.

**Avg. LRU Time**
Average amount of time taken by an LRU cleaner thread to clean a single LRU chain.

**Chunk Writes**
Number of changed buffers that were written to disk by a chunk cleaning operation. Chunk cleaning writes out all changed buffers of a given chunk that are in the buffer pool. This is done in a variety of special circumstances that need to clean a large number of buffers quickly, such as checkpoint processing and fast recovery.

**onstat -g cat: Print ER global catalog information**

**Syntax:**

```
>>>onstat -g cat
```

The **onstat -g cat** option prints information from the Enterprise Replication global catalog. The global catalog contains a summary of information about the defined servers, replicates, and replicate sets on each of the servers within the enterprise. If a replicated table is undergoing an alter operation, the **onstat -g cat** command shows that it is in alter mode. For example, use this command to determine:

- How many servers and how many replicates are configured
- Which table matches a given replicate
- Whether a server is a root or leaf server
- The current bitmap mask for a given server. You can use the bitmap mask with the output from the **onstat -g rqm** command to determine which server Enterprise Replication is waiting on for an acknowledgement.

The **onstat -g cat** command has the following formats:

- `onstat -g cat`
- `onstat -g cat scope`
- `onstat -g cat replname`

The following table describes `replname` and `scope`.

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>replname</code></td>
<td>The name of a replicate</td>
</tr>
<tr>
<td>scope</td>
<td>One of the following values:</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>servers</td>
<td>Print information on servers only</td>
</tr>
<tr>
<td>repls</td>
<td>Print information on replicates only</td>
</tr>
<tr>
<td>full</td>
<td>Print expanded information for both replicate servers and replicates</td>
</tr>
</tbody>
</table>

**Example Output:**

This sample output from the `onstat -g cat repls` command shows that the table `tab` is in alter mode. The replicate `rep1` is defined on this table, its replicate ID is 6553601. For more information on the replicate attributes that this command displays, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:01:39 -- 28672 Kbytes
GLOBAL-CATALOG CACHE STATISTICS
REPLICATES
-------------------
Parsed statements:
    Id 6553601 table tab
    Id 6553602 table tab12
Inuse databases: test(2)
   Name: rep1, Id: 6553601 State: ACTIVE Flags: 0x800000 ALTERMODE
       use 0 lastexec Wed Dec 31 18:00:00 1969
       Local Participant: test:nagaraju.tab
       Attributes: TXN scope, Enable ATS, Enable RIS, all columns
                   sent in updates
       Conflict resolution: [TIMESTAMP]
       Column Mapping: ON, columns INORDER, offset 8, uncomp_len 12
       Column Name Verification: ON
       No Replicated UDT Columns
Name: rep12, Id: 6553602 State: ACTIVE Flags: 0x800000 use 0
    lastexec Wed Dec 31 18:00:00 1969
    Local Participant: test:nagaraju.tab12
    Attributes: TXN scope, Enable ATS, Enable RIS, all columns
                sent in updates
    Conflict resolution: [TIMESTAMP]
    Column Mapping: ON, columns INORDER, offset 8, uncomp_len 2064
    Column Name Verification: ON
    No Replicated UDT Columns
```

*Figure 15-10. onstat -g cat repls Output*

**onstat -g cdr config: Print ER settings**

**Syntax:**

```
    -long
    -parameter_name
    -parameter_name long
    -CDR_ENV long
    -CDR_ENV variable_name
    -CDR_ENV variable_name long
```

The `onstat -g cdr config` option prints the settings of Enterprise Replication configuration parameters and environment variables that can be set with the CDR_ENV configuration parameter.
This command has the following formats:

- `onstat -g cdr config`  
- `onstat -g cdr config long`  
- `onstat -g cdr config parameter_name`  
- `onstat -g cdr config parameter_name long`  
- `onstat -g cdr config CDR_ENV`  
- `onstat -g cdr config CDR_ENV long`  
- `onstat -g cdr config CDR_ENV variable_name`  
- `onstat -g cdr config CDR_ENV variable_name long`

The `long` option prints additional information about settings that can be useful for IBM Support.

The following table describes `parameter_name` and `variable_name`:

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>parameter_name</code></td>
<td>The name of an Enterprise Replication configuration parameter</td>
</tr>
<tr>
<td><code>variable_name</code></td>
<td>The name of an Enterprise Replication environment variable</td>
</tr>
</tbody>
</table>

If you use `onstat -g cdr config` without any options, the settings of all Enterprise Replication configuration parameters and environment variables are included in the output. If you specify the CDR_ENV configuration parameter without an environment variable name, all Enterprise Replication environment variables are included in the output.

The following sample output of the `onstat -g cdr config ENCRYPT_CDR` command shows the setting of the ENCRYPT_CDR configuration parameter:

```
onstat -g cdr config ENCRYPT_CDR
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:06:17
ENCRIPT_CDR configuration setting: 0
```

The following sample output of the `onstat -g cdr config CDR_ENV` command shows the settings of all Enterprise Replication environment variables:

```
onstat -g cdr config CDR_ENV
CDR_ENV environment variable settings:
  CDR_LOGDELTA:
    CDR_LOGDELTA configuration setting: 0
  CDR_PERFLOG:
    CDR_PERFLOG configuration setting: 0
  CDR_ROUTER:
    CDR_ROUTER configuration setting: 0
  CDR_RMSCALEFACT:
    CDR_RMSCALEFACT configuration setting: 0
  CDRSITE_731:
    CDRSITE_731 configuration setting: [None configured]
  CDRSITE_92X:
    CDRSITE_92X configuration setting: [None configured]
  CDRSITE_10X:
    CDRSITE_10X configuration setting: [None configured]
```

The following sample output of the `onstat -g cdr config` command shows the settings of all Enterprise Replication configuration parameters and CDR_ENV environment variables:
onstat -g cdr config

IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:08:05

CDR_DBSPACE:
  CDR_DBSPACE configuration setting: rootdbs

CDR_DSLOCKWAIT:
  CDR_DSLOCKWAIT configuration setting: 5

CDR_EVALTHREADS:
  CDR_EVALTHREADS configuration setting: 1, 2

CDR_MAX_DYNAMIC_LOGS:
  CDR_MAX_DYNAMIC_LOGS configuration setting: 0

CDR_NIFCOMPRESS:
  CDR_NIFCOMPRESS configuration setting: 0

CDR_QDATA_SBSPACE:
  CDR_QDATA_SBSPACE configuration setting: cdrsbsp

CDR_QHDR_DBSPACE:
  CDR_QHDR_DBSPACE configuration setting: rootdbs

CDR_QUEUEMEM:
  CDR_QUEUEMEM configuration setting: 4096

CDR_SERIAL:
  CDR_SERIAL configuration setting: 0, 0

CDR_SUPPRESS_ATSRISWARN:
  CDR_SUPPRESS_ATSRISWARN configuration setting: [None suppressed]

ENCRYPT_CDR:
  ENCRYPT_CDR configuration setting: 0

ENCRYPT_CIPHERS:
  ENCRYPT_CIPHERS configuration setting: [None configured]

ENCRYPT_MAC:
  ENCRYPT_MAC configuration setting: [None configured]

ENCRYPT_MACFILE:
  ENCRYPT_MACFILE configuration setting: [None configured]

ENCRYPT_SWITCH:
  ENCRYPT_SWITCH configuration setting: 0, 0

CDR_ENV environment variable settings:

CDR_LOGDELTA:
  CDR_LOGDELTA configuration setting: 0

CDR_PERFLLOG:
  CDR_PERFLLOG configuration setting: 0

CDR_ROUTER:
  CDR_ROUTER configuration setting: 0

CDR_RMSCALEFACT:
  CDR_RMSCALEFACT configuration setting: 0

CDR_SITES_731:
  CDR_SITES_731 configuration setting: [None configured]

CDR_SITES_92X:
  CDR_SITES_92X configuration setting: [None configured]

CDR_SITES_10X:
  CDR_SITES_10X configuration setting: [None configured]

---

onstat -g con: Print condition and thread information

Syntax:

```
>onstat -g con
```

The onstat -g con option prints information on conditions and the threads that are waiting for them.

Example Output:
Figure 15-11. onstat -g con Output

Output Description:

<table>
<thead>
<tr>
<th>cid</th>
<th>addr</th>
<th>name</th>
<th>waiter</th>
<th>waittime</th>
</tr>
</thead>
<tbody>
<tr>
<td>271</td>
<td>c63d930</td>
<td>netnorm</td>
<td>1511</td>
<td>6550</td>
</tr>
</tbody>
</table>

**onstat -g ddr: Print ER database log reader status**

**Syntax:**

```
$ onstat -g ddr
```

The **onstat -g ddr** option prints the status of the Enterprise Replication database log reader. The **ddr**, or **ddr_snoopy**, is an internal component of Enterprise Replication that reads the log buffers and passes information to the grouper.

You can use the information from the **onstat -g ddr** command to monitor **replay position** in the log file and ensure replay position is never overwritten (which can cause loss of data). The replay position is the point from where, if a system failure occurs, Enterprise Replication starts re-reading the log information into the log update buffers. All the transactions generated before this position at all the target servers have been applied by Enterprise Replication or safely stored in stable queue space.

The **onstat -g ddr** output shows you a snapshot of the replay position, the **snoopy position**, and the **current position**. The snoopy position identifies the position of the **ddr_snoopy** thread in the logical logs. The **ddr_snoopy** has read the log records up until this point. The current position is the position where the server has written its last logical log record.

The **log needs** position is based on replay position and is set at a certain distance from replay position, for example, at seventy percent of the log file. The remainder of the circular log file comprises the DDR BLOCK zone. As messages are acknowledged or stored in the stable queue, the replay position, and hence also the log needs position, should advance. If you notice that replay position is not advancing, this can mean that the stable queue is full or a remote server is down.

If log reading is blocked, data might not be replicated until the problem is resolved. If the block is not resolved, the database server might overwrite the read (**ddr_snoopy**) position, which means that data will not be replicated. If this occurs, you must manually resynchronize the source and target databases.
For servers of Version 9.4, and later, you can enable dynamic log creation by setting the CDR_MAX_DYNAMIC_LOGS configuration parameter in the ONCONFIG file. If the current position reaches the log needs position, instead of going into a blocked state, Enterprise Replication automatically adds another log file. If this option is set, the onstat -g ddr command prints the number of dynamic log requests made. For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

**Example Output:**

The following sample output from the onstat ddr command shows the replay position, snoopy position, and current position highlighted.

```
   DDR -- Running
   # Event Snoopy Snoopy Replay Replay Current Current
   Buffers ID Position ID Position ID Position
   528 24 165018 24 6a018 24 166000

   Log Pages Snooped: From From Tossed
   Cache Disk (LBC full)
   247 111 0

   Total dynamic log requests: 0
   DDR events queue

   Type TX id Partnum Row id
```

*Figure 15-12. onstat -g ddr Output*

**onstat -g dic: Print table information**

**Syntax:**

```
   onstat -g dic
```

Without any parameters the onstat -g dic option prints one line of information for each table cached in the shared-memory dictionary. If a table name is specified, prints internal SQL information for that table.

For more information see the *Performance Guide*.

**Example Output:**
-- 101376 Kbytes

Dictionary Cache:  Number of lists: 31, Maximum list size: 10

<table>
<thead>
<tr>
<th>list#</th>
<th>size</th>
<th>refcnt</th>
<th>dirty?</th>
<th>heapptr</th>
<th>table name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>no</td>
<td>14b5d890</td>
<td>wbe@oninit_shm:informix.t0010url</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14cb820</td>
<td>wbe@oninit_shm:informix.t9051themeval</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14b63c20</td>
<td>wbe@oninit_shm:informix.t0060hits</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>no</td>
<td>14b97420</td>
<td>wbe@oninit_shm:informix.t0120import</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14b6c820</td>
<td>wbe@oninit_shm:informix.t9110domain</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0</td>
<td>no</td>
<td>14bce020</td>
<td>wbe@oninit_shm:informix.t0150url</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14d3d020</td>
<td>contact@oninit_shm:informix.wbtags</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14c87420</td>
<td>wbe@oninit_shm:informix.wbtags</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>no</td>
<td>14b7a420</td>
<td>drug@oninit_shm:viagra.product</td>
</tr>
</tbody>
</table>

Total number of dictionary entries: 36

**Figure 15-13. onstat -g dic Output**

### Output Description:
- **list#**: Data dictionary hash chain ID
- **size**: Number of entries in this hash
- **refcnt**: Number of SQL statements currently referencing one of the cache entries.
- **dirty?**: Whether the entry has been modified since last written to disk.
- **heapptr**: Address for the heap used to store this table
- **table name**: Name of table in cache

**onstat -g dis: Print database server information**

**Syntax:**
```
>>---onstat--- -g--dis---
```

Prints a list of database servers and their status, and information about each database server, `INFORMIXDIR`, `sqlhosts` file, `ONCONFIG` file, and hostname. You can use this option with the database server in any mode, including offline.

**Example Output:**
-- 101376 Kbytes

There are 2 servers found
Server : ol_tuxedo
Server Number : 53
Server Type : IDS
Server Status : Up
Server Version: IBM Informix Dynamic Server Version 11.10.UC1
Shared Memory : 0xa000000
INFORMIXDIR : /local1/engines/ol_tuxedo/dist
ONCONFIG : /local1/engines/ol_tuxedo/dist/etc/onconfig.ol_tuxedo
SQLHOSTS : /local1/engines/ol_tuxedo/dist/etc/sqlhosts
Host : avocet

Server : ol_9next
Server Number : 0
Server Type : IDS
Server Status : Down
Server Version: IBM Informix Dynamic Server Version 11.10.UC1
Shared Memory : 0
INFORMIXDIR : /local1/engines/ol_9next/dist
ONCONFIG :
SQLHOSTS :
Host :

Figure 15-14. onstat -g dis Output

**Output Description:**
- **Server** Server name
- **Server Number** Number of the server.
- **Server Type** Type of server
- **Server Status** Up means that the server is online, Down means that the server is offline
- **Server Version** Version of the server
- **Shared Memory** Location of the shared memory address
- **INFORMIXDIR** Location of the $INFORMIXDIR/ directory on UNIX and in the %INFORMIXDIR%\ directory on Windows.
- **ONCONFIG** Location of the ONCONFIG file
- **SQLHOSTS** Location of the sqlhosts file
- **Host** Host name of the server

**onstat -g dll: Print dynamic libraries list**

**Syntax:**
```
>>>onstat -g dll
```

Prints a list of dynamic libraries that have been loaded.

**Example Output:**

Datablades:
<table>
<thead>
<tr>
<th>addr</th>
<th>slot</th>
<th>vp</th>
<th>baseaddr</th>
<th>filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>140090fc</td>
<td>2</td>
<td>1</td>
<td>fe644d4e</td>
<td>MYPATH/informix/extend/web.xxxxx/web.bld</td>
</tr>
<tr>
<td>141c70fc</td>
<td>2</td>
<td>1</td>
<td>fe7cd4e0</td>
<td></td>
</tr>
<tr>
<td>141ca0fc</td>
<td>3</td>
<td>1</td>
<td>fe7cd4e0</td>
<td></td>
</tr>
</tbody>
</table>

Figure 15-15. onstat -g dll Output

Output Description:
- **addr**: DLL address
- **slot**: Slot number entry in the library table
- **vp**: Virtual processor ID
- **baseaddr**: Virtual processor base address
- **filename**: DLL filename

**onstat -g dmp**: Print raw memory

Syntax:

```
> onstat -g dmp
```

The **onstat -g dmp** option prints raw memory at a given address for a number of given bytes. Each address and length must be within the allocated memory shown from **onstat -g seg** output.

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:03:58 -- 1058816 Kbytes

address     bytes in mem
0000000010a00000: ee070001 0000120e ffffffff 00000000 ........ ........
0000000010a00010: 00000000 00000000 00000000 00000000 ........ ........
0000000010a00020: 00000000 00000000 00000000 00000000 ........ ........
0000000010a00030: 00000000 00000000 00000000 00000000 ........ ........
0000000010a00040: 00000000 00000000 00000000 00000000 ........ ........
0000000010a00050: 00000000 00000000 00000000 00000000 ........ ........
0000000010a00060: 00000000
```

Figure 15-16. onstat -g dmp Output

Output Description:
- **address**: Memory address of the raw memory
- **bytes in mem**: Hexadecimal and ASCII representations of the memory contents
onstat -g dri: Print HDR information

Syntax:

```
$ onstat -g dri
```

The onstat -g dri option prints information about High-Availability Data Replication on the current server.

Example Output:

```
Data Replication:
Type   State   Paired server   Last DR CKPT (id/pg)
primary  off    amit_secondary    -1 / -1

DRINTERVAL 2
DRTIMEOUT 30
DRAUTO 0
DRLOSTFOUND /vobs/tristarm/sqldist/etc/dr.lostfound
DRIDXAUTO 0
```

Figure 15-17. onstat -g dri Output

Output Description:

- **Type**: Current type of server: primary, secondary, or standard
- **State**: on or off
- **Paired server**: Name of the primary or secondary server that this server is paired with
- **Last DR CKPT**: Last checkpoint ID and page

The second section lists the values of the following configuration parameters in the ONCONFIG file:

- DRINTERVAL
- DRTIMEOUT
- DRAUTO
- DRLOSTFOUND
- DRIDXAUTO

onstat -g dsc: Print distribution cache information

Syntax:

```
$ onstat -g dsc
```

Prints a list of distribution cache information.

Example Output:
Distribution Cache:

- Number of lists: 31
- PC_POOLSIZE: 50
- Number of entries: 0
- Number of entries in use: 0

Distribution Cache Entries:

<table>
<thead>
<tr>
<th>list#</th>
<th>id</th>
<th>ref_cnt</th>
<th>dropped?</th>
<th>heap_ptr</th>
<th>distribution name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Distribution Cache is empty.

**Figure 15-18. onstat -g dsc Output**

**Output Description:**

The first section of output describes the distribution cache.

- **Number of lists**: Number of lists in the distribution cache
- **PC_POOLSIZE**: Number of entries that can be cached at one time
- **Number of entries**: Number of entries in the distribution cache
- **Number of entries in use**: Number of entries being used

The second section of output describes the distribution cache entries.

- **list#**: Distribution cache hash chain ID
- **id**: Number of hash entries
- **ref_cnt**: Number of statements referencing a cache entry
- **dropped?**: Whether this entry has been dropped since being added to the cache
- **heap_ptr**: Heap address used to store this entry
- **distribution name**: The name of the distribution in the cache

**onstat -g dss: Print ER environment data**

**Syntax:**

```
$ onstat -g dss [modifier]
```

The **onstat -g dss** command prints detailed statistical information about the activity of individual data sync threads in an Enterprise Replication environment. The data sync thread applies the transaction on the target server. Statistics include the number of applied transactions and failures and when the last transaction from a source was applied.

The **onstat -g dss** command has the following formats:

- onstat -g dss
- onstat -g dss modifier

---

**Chapter 15. The onstat Utility** 15-39
The following table describes the values for modifier.

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDR</td>
<td>Prints summary information about any UDR invocations by the data sync threads.</td>
</tr>
<tr>
<td>UDRx</td>
<td>Prints expanded information (including a summary of error information) about any UDR invocations by the data sync threads. The ProcId column lists the UDR procedure ID.</td>
</tr>
</tbody>
</table>

Example Output:

In the following example, only one data sync thread is currently processing the replicated data. It has applied a total of one replicated transaction and the transaction was applied at 2004/09/13 18:13:10. The Processed Time field shows the time when the last transaction was processed by this data sync thread.

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:00:28 -- 28672 Kbytes
DS thread statistic
  cmtTime         Tx  .putExtra Tx Last Tx
  Name     < local Committed Aborted Processed Processed Time
  -----------------------------------------------
  CDRD_1     0       1       0       1   (1095117190) 2004/09/13 18:13:10

Tables (0.0%): Databases: test
  CDR_DSLOCKWAIT = 1
  CDR_DSCLOSEINTERVAL = 60
```

Figure 15-19. onstat -g dss Output

**onstat -g dtc: Print delete table cleaner statistics**

**Syntax:**

```
'onstat -g dtc'
```

The `onstat -g dtc` command prints statistics about the delete table cleaner. The delete table cleaner removes rows from the delete table when they are no longer needed.

The `-g dtc` option is used primarily as a debugging tool and by IBM Support.

Example Output:

In the following example, the thread name of the delete table cleaner is **CDRDTCleaner**. The total number of rows deleted is 1. The last activity on this thread occurred at 2006/09/13 18:47:19. The delete table for replicate **rep1** was last cleaned at 2006/09/13 18:28:25.
onstat -g env: Prints environment variable values

Syntax:

```bash
onstat -g env [sessionid] [variable]
```

The `onstat -g env` option displays the values of environment variables the database server currently uses. You can specify one of the following invocations.

<table>
<thead>
<tr>
<th>Invocation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>onstat -g env</code></td>
<td>Displays the settings of variables when the database server was started.</td>
</tr>
<tr>
<td></td>
<td>Does not display variables that have not been set explicitly.</td>
</tr>
<tr>
<td><code>onstat -g env sessionid</code></td>
<td>Displays the settings that a specific session uses. This display includes</td>
</tr>
<tr>
<td></td>
<td>the following values:</td>
</tr>
<tr>
<td></td>
<td>• Set in the environment of the session</td>
</tr>
<tr>
<td></td>
<td>• Assigned by the database server, as <code>onstat -g env</code> displays</td>
</tr>
<tr>
<td><code>onstat -g env all</code></td>
<td>Displays the settings used by all sessions</td>
</tr>
<tr>
<td></td>
<td>This display is the same as the output of <code>onstat -g env</code> and</td>
</tr>
<tr>
<td></td>
<td><code>onstat -g env sessionid</code> iteratively on all current sessions.</td>
</tr>
<tr>
<td><code>onstat -g env variable</code></td>
<td>Displays the default value of the specified variable</td>
</tr>
<tr>
<td></td>
<td>This <code>variable</code> argument eliminates the need to pipe the output to <code>grep</code></td>
</tr>
<tr>
<td></td>
<td>(or some other utility) to locate a variable among many that might be set.</td>
</tr>
<tr>
<td><code>onstat -g env sessionid variable</code></td>
<td>Displays the value of the specified variable that the specified session</td>
</tr>
<tr>
<td></td>
<td>uses</td>
</tr>
<tr>
<td></td>
<td>The <code>sessionid</code> and <code>variable</code> arguments eliminate the need to pipe the</td>
</tr>
<tr>
<td></td>
<td>output to <code>grep</code> (or some other utility) to locate a variable among many that</td>
</tr>
<tr>
<td></td>
<td>might be set.</td>
</tr>
</tbody>
</table>

Figure 15-20. onstat -g dtc Output
You might want to display the values of environment variables in any of the following situations:

- The database server instance has been up for months, and you cannot remember the setting of an environment variable (such as the server locale setting \texttt{SERVER\_LOCALE}).
- You want to display the complete list of values for a variable to identify when a variable has been set in multiple places.
- Environment files on disk might have changed or been lost in the interim.
- A support engineer wants to know settings of specific environment variables.

The \texttt{onstat -g env} option displays the current setting of a variable and the complete list of values each time the variable was set in the environment. For example, if \texttt{PDQPRIORITY} is set to 10 in the \texttt{.informix.rc} file and set to 55 in the shell environment, \texttt{onstat -g env} displays both values.

However, if you change the \texttt{PDQPRIORITY} with the \texttt{onmode -q pdqpriority sessionid} option, \texttt{onstat -g env} does not display the new value for the session. The \texttt{onstat -g env} option displays only the values of variables set in the environment. It does not display values modified while the session is running.

The following figure show the output for the \texttt{onstat -g env} option
onstat -g ffr: Print free fragments

Syntax:

```plaintext
'onstat -g ffr [name session id]
```

The `onstat -g ffr` option prints free fragments for a pool of shared memory.

Example Output:
Output Description:

addr Pool fragment address
size Fragment size, in bytes

onstat -g glo: Print global multithreading information

Syntax:

```
> onstat -g glo
```

The onstat -g glo option prints global multithreading information.

Example Output:

<table>
<thead>
<tr>
<th>addr</th>
<th>size</th>
</tr>
</thead>
<tbody>
<tr>
<td>10ac92f20</td>
<td>224</td>
</tr>
<tr>
<td>10ac8c8c0</td>
<td>3904</td>
</tr>
</tbody>
</table>

IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 01:55:02 -- 101376 Kbytes

MT global info:
sessions  threads  vps  lngspins
0  49  14  1

sched calls  thread switches  yield 0  yield n  yield forever
total: 900100  898846  1238  27763  423778

per sec: 327  325  2  12  151

Virtual processor summary:
class  vps  usercpu  syscpu  total
cpu   4  0.92  0.10  1.02
aio   4  0.02  0.02  0.04
lio   1  0.00  0.00  0.00
pio   1  0.00  0.00  0.00
adm   1  0.00  0.01  0.01
msc   1  0.00  0.00  0.00
fifo  2  0.00  0.00  0.00
tot 14  0.94  0.13  1.07

Individual virtual processors:

<table>
<thead>
<tr>
<th>vp</th>
<th>pid</th>
<th>class</th>
<th>usercpu</th>
<th>syscpu</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2599</td>
<td>cpu</td>
<td>0.25</td>
<td>0.06</td>
<td>0.31</td>
</tr>
<tr>
<td>2</td>
<td>2602</td>
<td>adm</td>
<td>0.00</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>3</td>
<td>2603</td>
<td>cpu</td>
<td>0.23</td>
<td>0.00</td>
<td>0.23</td>
</tr>
<tr>
<td>4</td>
<td>2604</td>
<td>cpu</td>
<td>0.21</td>
<td>0.03</td>
<td>0.24</td>
</tr>
<tr>
<td>5</td>
<td>2605</td>
<td>cpu</td>
<td>0.23</td>
<td>0.01</td>
<td>0.24</td>
</tr>
<tr>
<td>6</td>
<td>2606</td>
<td>lio</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>7</td>
<td>2607</td>
<td>pio</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>8</td>
<td>2608</td>
<td>aio</td>
<td>0.02</td>
<td>0.02</td>
<td>0.04</td>
</tr>
<tr>
<td>9</td>
<td>2609</td>
<td>msc</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>10</td>
<td>2610</td>
<td>fifo</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>11</td>
<td>2611</td>
<td>fifo</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>12</td>
<td>2612</td>
<td>aio</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>13</td>
<td>2613</td>
<td>aio</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>14</td>
<td>2614</td>
<td>aio</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>tot</td>
<td></td>
<td>0.94</td>
<td>0.13</td>
<td>1.07</td>
<td></td>
</tr>
</tbody>
</table>

Figure 15-22. onstat -g ffr Output

Figure 15-23. onstat -g glo Output
Output Description:

Virtual Processor Summary

- **class**: The type of virtual processor
- **vps**: The number of instances of this class of VP
- **usercpu**: The total user time this class of VP has spent running on the CPU in seconds
- **syscpu**: The total system time this class of VP has spent running on the CPU in seconds
- **total**: The total number of virtual processors, user time and system time

Individual Virtual Processors

- **vp**: Virtual processor number
- **pid**: Process ID of this oninit process
- **class**: Virtual processor class
- **usercpu**: Total user time the VP has run on the CPU
- **syscpu**: Total system time the VP has run on the CPU
- **total**: Total number of VPs, user time, and system time

onstat -g grp: Print ER grouper statistics

**Syntax:**

```
  onstat -g grp [modifier]
```

The `onstat -g grp` command prints statistics about the Enterprise Replication grouper. The grouper evaluates the log records, rebuilds the individual log records into the original transaction, packages the transaction, and queues the transaction for transmission.

The `-g grp` option is used primarily as a debugging tool and by Technical Support.

The onstat -g grp command has the following formats:
- `onstat -g grp`
- `onstat -g grp modifier`

The following table describes the values for `modifier`.

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Prints all the information printed by the G, T, P, E, R, and S modifiers</td>
</tr>
<tr>
<td>E</td>
<td>Prints grouper evaluator statistics</td>
</tr>
<tr>
<td>Ex</td>
<td>Prints grouper evaluator statistics, expands user-defined routine (UDR) environments</td>
</tr>
<tr>
<td>G</td>
<td>Prints grouper general statistics</td>
</tr>
<tr>
<td>L</td>
<td>Prints grouper global list</td>
</tr>
<tr>
<td>Lx</td>
<td>Prints grouper global list, expands open transactions</td>
</tr>
<tr>
<td>M</td>
<td>Prints grouper compression statistics</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>Mz</td>
<td>Clears grouper compression statistics</td>
</tr>
<tr>
<td>P</td>
<td>Prints grouper table partition statistics</td>
</tr>
<tr>
<td>pager</td>
<td>Prints grouper paging statistics</td>
</tr>
<tr>
<td>R</td>
<td>Prints grouper replicate statistics</td>
</tr>
<tr>
<td>S</td>
<td>Prints grouper serial list head (The serial list head is the first transaction in the list, that is, the next transaction that will be placed in the send queue.)</td>
</tr>
<tr>
<td>SI</td>
<td>Prints grouper serial list (The serial list is the list of transactions, in chronological order.)</td>
</tr>
<tr>
<td>Sx</td>
<td>Prints grouper serial list, expands open transactions</td>
</tr>
<tr>
<td>T</td>
<td>Prints grouper transaction statistics</td>
</tr>
<tr>
<td>UDR</td>
<td>Prints summary information about any UDR invocations by the grouper threads</td>
</tr>
<tr>
<td>UDRx</td>
<td>Prints expanded information (including a summary of error information) about any UDR invocations by the grouper threads The Proc id column lists the UDR procedure ID.</td>
</tr>
</tbody>
</table>

**Example Output:**

This section contains sample output from various `onstat -g grp modifier` commands. The following sample shows output for the `onstat -g grp` command.

```
Grouper at 0xb014018:
  Last Idle Time: (1095122236) 2004/09/13 19:37:16
  RSAM interface ring buffer size: 528
  RSAM interface ring buffer pending entries: 0
  Eval thread interface ring buffer size: 48
  Eval thread interface ring buffer pending entries: 0
  Log update buffers in use: 0
  Max log update buffers used at once: 5
  Log update buffer memory in use: 0
  Max log update buffer memory used at once: 320
  Updates from Log: 16
  Log update links allocated: 512
  Blob links allocated: 0
  Conflict Resolution Blocks Allocated: 0
  Memory pool cache: Empty
  Last Tx to Queue began : (1095118105) 2004/09/13 18:28:25
  Last Tx to Queue ended : (1095118105) 2004/09/13 18:28:25
  Last Tx to Queue log ID, position: 12,23
  Open Tx: 0
  Serial Tx: 0
  Tx not sent: 0
  Tx sent to Queue: 2
  Tx returned from Queue: 2
  Events sent to Queue: 7
  Events returned from Queue: 7
  Total rows sent to Queue: 2
  Open Tx array size: 1024
  Table 'tab' at 0xae8ebb0 [ CDRShadow ]
  Table 'tab12' at 0xae445e0 [ CDRShadow ]
```

*Figure 15-24. onstat -g grp Output (Part 1 of 3)*
The following example shows output for the `onstat -g grp E` command. The field **Evaluators**: 4 indicates that there are four evaluation threads configured for the system.
The following example shows output for the `onstat -g grp G` command.

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 02:07:10 -- 36864 Kbytes
Repl links on global free list: 0 Evaluators: 4
Evaluator at 0xba71840 ID 0 [Idle:Idle] Protection: unused
   Eval iteration: 1007
   Updates evaluated: 0
   Repl links on local free list: 256
   UDR environment table at 0xba71890
      Number of environments: 0
      Table memory limit : 16777
      Table memory used : 0
      API memory limit : 131072
      API memory used : 0
      Count failed UDR calls: 0
Evaluator at 0xba718f0 ID 1 [Idle:Idle] Protection: unused
   Eval iteration: 1007
   Updates evaluated: 0
   Repl links on local free list: 256
   UDR environment table at 0xba71940
      Number of environments: 0
      Table memory limit : 16777
      Table memory used : 0
      API memory limit : 131072
      API memory used : 0
      Count failed UDR calls: 0
Evaluator at 0xba8c260 ID 2 [Idle:Idle] Protection: unused
   Eval iteration: 1007
   Updates evaluated: 0
   Repl links on local free list: 256
   UDR environment table at 0xba8c2b0
      Number of environments: 0
      Table memory limit : 16777
      Table memory used : 0
      API memory limit : 131072
      API memory used : 0
      Count failed UDR calls: 0
Evaluator at 0xbaac2a0 ID 3 [Idle:Idle] Protection: unused
   Eval iteration: 1007
   Updates evaluated: 0
   Repl links on local free list: 256
   UDR environment table at 0xbaac2f0
      Number of environments: 0
      Table memory limit : 16777
      Table memory used : 0
      API memory limit : 131072
      API memory used : 0
      Count failed UDR calls: 0
Total Free Repl links 1024
```

Figure 15-25. onstat -g grp E Output (Part 1 of 2)
The following example shows output for the `onstat -g grp P` command. In the example, the grouper is evaluating rows for the `account`, `teller` and `customer` tables.

```
Table 'teller' at 0xb851480 [CDRShadow VarChars ]
Table 'account' at 0xb7faad8 [CDRShadow VarChars VarUDTs Floats Blobs]
Table 'customer' at 0xbbe67a8 [CDRShadow VarChars VarUDTs]
Grouper Table Partitions:
  Slot 387... 'account' 1048707
  Slot 389... 'teller' 1048709
  Slot 394... 'customer' 1048714
```

Figure 15-27. onstat -g grp P Output

The following example shows output for the `onstat -g grp pager` command. The sample output shows the grouper large transaction evaluation statistics.

```
IBM Informix Dynamic Server Version 11.10.UC1  -- On-Line -- Up 00:20:42 -- 28672 Kbytes
Grouper Pager statistics:
Number of active big transactions: 0
Total number of big transactions processed: 0
Spool size of the biggest transaction processed: 0 Bytes
```

Figure 15-28. onstat -g grp pager Output

The following example shows output for the `onstat -g grp R` command. In this example, the grouper is configured to evaluate rows for replicates with IDs 6553601 and 6553602 (you can use the `onstat -g cat repls` command to obtain the replicate names). The `Ignore` attribute of replicate ID 6553602 shows that the grouper is currently not evaluating rows for this replicate. This can happen if the replicate state is not ACTIVE. You can obtain the replicate state using the `onstat -g cat repls` command.
The following example shows output for the `onstat -g grp T` command. In this example, the grouper evaluated and queued 1 transaction to the send queue. The `Tx sent to Queuer` field shows the total number of transactions evaluated and queued to the send queue for propagating to all the replicate participants. The `Total rows sent to Queuer` field shows the total number of rows queued to the send queue for propagating to all the replicate participants.

```
Figure 15-29. onstat -g grp R Output

IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:04:47 -- 28672 Kbytes
Replication Group 6553601 at 0xb0a8360
  Replication at 0xb0a82b0 6553601:6553601 (tab) [ NotifyDS FullRowOn ]
    Column Information [ CDRShadow VarUDTs InOrder Same ]
    CDR Shadow: offset 0, size 8
    In Order: offset 8, size 10
Replication Group 6553602 at 0xb0a8480
  Replication at 0xb0a83d0 6553602:6553602 (tab12) [ Ignore Stopped NotifyDS FullRowOn ]
    Column Information [ CDRShadow VarUDTs InOrder Same ]
    CDR Shadow: offset 0, size 8
    In Order: offset 8, size 16
```

```
Figure 15-30. onstat -g grp T Output

IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:14:51 -- 28672 Kbytes
Last Tx to Queue began : (1095116676) 2004/09/13 18:04:36
Last Tx to Queue ended : (1095116676) 2004/09/13 18:04:36
Last Tx to Queue log ID, position: 5,3236032
Open  Tx: 0
Serial Tx: 0
Tx not sent: 0
Tx sent to Queue: 1
  Tx returned from Queue: 0
  Events sent to Queue: 0
  Events returned from Queue: 0
  Total rows sent to Queue: 1
  Open Tx array size: 1024
```

**onstat -g his: Print SQLTRACE information**

**Syntax:**

```
    onstat -g his
```

The `onstat -g his` option prints information about the SQLTRACE configuration parameter. By default, only the DBSA can view `onstat -g his` syssqltrace information. However, when `UNSECURE_ONSTAT = 1` all users can view this information.

**Example Output:**
| Statement Statistics: | | | | | | | |
|----------------------|------------------------|------------------------|------------------------|------------------------|------------------------|------------------------|
| Page                 | Buffer Read            | % Cache                | Buffer Page            | Buffer Write           | % Cache                | |
| Read                 | Read                   | 1285                   | 19444                  | 93.39                  | 5359                   | 810                    |
|                      |                        |                        |                        |                        |                        |                        |
| Lock                 | Lock                   | Lock                   | Log                    | Num                    | Disk                   | Memory                 |
| Requests             | Waits                  | Total                  | Avg                    | Max                    | I/O Wait               | Avg Rows               |
| 10603                | 0                      | 0                      | 0                      | 0                      | 60.4 KB                | 0                      |
|                      |                        |                        |                        |                        |                        |                        |
| Estimated Cost       | Estimated SQL          | Actual SQL             | ISAM                   | Isolation              | SQL                    | |
| 102                  | 1376                   | 5244                   | 0                      | 0                      | CR                     | 32608                  |

**Figure 15-31. onstat -g his Output**

**Output Description:**

**Page Read**
Number of pages that have been read from disk

**Buffer Reads**
Number of times a page has been read from the buffer pool and not read from disk

**Read % Cache**
Percentage of times the page should be read from the buffer pool

**Buffer IDX Read**
Number of buffer reads for index pages

**Page Write**
Number of pages written to disk

**Buffer Write**
Number of pages modified and sent back to the buffer pool

**Write % Cache**
Percentage of time that a page was written to the buffer pool but not to disk

**Lock Requests**
Total number of locks required by this statement

**Lock Waits**
Number of times this SQL statement waited on locks

**LK Wait Time**
Time spent waiting for locks during this SQL statement in seconds

**Log Space**

**Num Sorts**
Total number of sorts used to execute the sort

**Disk Sorts**
Number of sorts for this SQL statement that were executed on disk

**Memory Sorts**
Total Executions
Total number of times this statement has been executed or the number of
times this cursor has been re-used

Total Time
Total time executing this statement in seconds

Avg Time
Average time this state takes to execute in seconds

Max Time
Total time to run the SQL statement in seconds, excluding any time taken
by the application

LK Wait Time
Amount of time the statement waited for application locks

Avg IO Wait
Amount of time the statement waited for I/O, excluding any asynchronous
I/O.

Avg Rows Per Sec
Average number of rows a second produced by this statement

Estimated Cost
Cost associated with the SQL statement

Estimated Rows
Estimated number of rows returned, as estimated by the optimizer for the
statement

Actual Rows
Number of rows returned for this statement

SQL Error
The SQL error number

ISAM Error
The RSAM/ISAM error number

Isolation Level
Isolation level this statement was run with

SQL Memory
Number of bytes this SQL statement requires

onstat -g ioa: Print combined onstat -g information

Syntax:
-- onstat -g ioa --

The onstat -g ioa option prints combined information from -g ioq, -g iov, and -g
iob.

Example Output:
For a description of each output column, see the -g ioq, -g iov, and -g iob options.

**onstat -g iob: Print big buffer use summary**

**Syntax:**

```
>>> onstat -g iob
```

The `onstat -g iob` option prints a summary of big buffer use.

**Example Output:**

---

**onstat -g iob Output**

Output Description:

For a description of each output column, see the -g ioq, -g iov, and -g iob options.
onstat -g iof: Print asynchronous I/O statistics

Syntax:

```bash
onstat -g iof
```

The `onstat -g iof` option prints asynchronous I/O statistics by chunk or file. This option is similar to the `-D` option, except that information on nonchunk files is also displayed. It includes information about temporary files and sort-work files.

Example Output:

```
AIO big buffer usage summary:
class  reads  writes
   fifo    pages  ops  pgs/op  holes  hl-ops  hls/op
      0        0   0.00       0      0   0.00   0   0   0.00
  kio    0        0   0.00       0      0   0.00   0   0   0.00
  adt    0        0   0.00       0      0   0.00   0   0   0.00
  msc    0        0   0.00       0      0   0.00   0   0   0.00
  aio    0        0   0.00       0      0   0.00   0   0   0.00
  pio    0        0   0.00       0      0   0.00   0   0   0.00
  lio    0        0   0.00       0      0   0.00   0   0   0.00
```

```
AIO global files:
gfd pathname  totalops  dskread  dskwrite  io/s
  3 rootdbs.1   613        0    613   0.3
```

Output Description:

- `gfd`: Global file descriptor number for this chunk
- `pathname`: The pathname of the chunk
- `totalops`: Total number of read and write operations that have occurred against the chunk
- `dskread`: Number of disk read that have occurred against the chunk
- `dskwrite`: Number of disk writes that have occurred against the chunk
- `io/s`: Number of I/Os per second
onstat -g iog: Print AIO global information

Syntax:

```
onstat -g iog```

The `onstat -g iog` option prints AIO global information.

Example Output:

```
AIO global info:
  7 aio classes
  4 open files
  64 max global files
32768 max files from setrlimit
```

Figure 15-35. onstat -g iog Output

onstat -g ioq: Print I/O queue information

Syntax:

```
onstat -g ioq [queue_name]```

The `onstat -g ioq` option shows statistics about the number and types of operations performed by I/O queues. If a `queue_name` is given then only queues with that name are shown. If no `queue_name` is given then information is given for all queues.

Example Output:
**Output Description:**

<table>
<thead>
<tr>
<th>q name/id</th>
<th>len</th>
<th>maxlen</th>
<th>totalops</th>
<th>dskread</th>
<th>dskwrite</th>
<th>dskcopy</th>
</tr>
</thead>
<tbody>
<tr>
<td>sql_dbg</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>fifo</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>adt</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>537</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>msc</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>6537</td>
<td>238</td>
<td>5777</td>
</tr>
<tr>
<td>aio</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1103</td>
<td>0</td>
<td>1102</td>
</tr>
<tr>
<td>pio</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>11795</td>
<td>0</td>
<td>11794</td>
</tr>
<tr>
<td>lio</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>17489</td>
<td>1526</td>
<td>15963</td>
</tr>
<tr>
<td>gfd 3</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>18347</td>
<td>2384</td>
<td>15963</td>
</tr>
<tr>
<td>gfd 4</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>220</td>
<td>41</td>
<td>179</td>
</tr>
<tr>
<td>gfd 5</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>gfd 6</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>gfd 7</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>gfd 8</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>54</td>
<td>24</td>
<td>30</td>
</tr>
<tr>
<td>gfd 9</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>149</td>
<td>40</td>
<td>109</td>
</tr>
<tr>
<td>gfd 10</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>621</td>
<td>128</td>
<td>493</td>
</tr>
<tr>
<td>gfd 11</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>753</td>
<td>1146</td>
<td>807</td>
</tr>
<tr>
<td>gfd 12</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>409</td>
<td>71</td>
<td>338</td>
</tr>
<tr>
<td>gfd 13</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>378</td>
<td>60</td>
<td>318</td>
</tr>
<tr>
<td>gfd 14</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>378</td>
<td>60</td>
<td>318</td>
</tr>
</tbody>
</table>

**Figure 15-36. onstat -g iq Output**

<table>
<thead>
<tr>
<th>q name/id</th>
<th>len</th>
<th>maxlen</th>
<th>totalops</th>
<th>dskread</th>
<th>dskwrite</th>
<th>dskcopy</th>
</tr>
</thead>
<tbody>
<tr>
<td>sql_dbg</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>fifo</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>adt</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>537</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>msc</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>6537</td>
<td>238</td>
<td>5777</td>
</tr>
<tr>
<td>aio</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1103</td>
<td>0</td>
<td>1102</td>
</tr>
<tr>
<td>pio</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>11795</td>
<td>0</td>
<td>11794</td>
</tr>
<tr>
<td>lio</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>17489</td>
<td>1526</td>
<td>15963</td>
</tr>
<tr>
<td>gfd 3</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>18347</td>
<td>2384</td>
<td>15963</td>
</tr>
<tr>
<td>gfd 4</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>220</td>
<td>41</td>
<td>179</td>
</tr>
<tr>
<td>gfd 5</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>gfd 6</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>gfd 7</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>gfd 8</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>54</td>
<td>24</td>
<td>30</td>
</tr>
<tr>
<td>gfd 9</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>149</td>
<td>40</td>
<td>109</td>
</tr>
<tr>
<td>gfd 10</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>621</td>
<td>128</td>
<td>493</td>
</tr>
<tr>
<td>gfd 11</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>753</td>
<td>1146</td>
<td>807</td>
</tr>
<tr>
<td>gfd 12</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>409</td>
<td>71</td>
<td>338</td>
</tr>
<tr>
<td>gfd 13</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>378</td>
<td>60</td>
<td>318</td>
</tr>
</tbody>
</table>

**Output Description:**

- **q name/id**: The name and number of the I/O queue. The name indicates what type of queue it is. The number is used to tell queues of the same name apart.

- **Here is a list of the possible queue names and what each type of queue handles:**

  - **sql_dbg**: Handles I/O for IBM Technical Support’s SQL Interface Debugging feature
  - **fifo**: Handles I/O for FIFO VPs
  - **adt**: Handles auditing I/O
  - **msc**: Handles miscellaneous I/O
  - **aio**: Handles IBM Informix asynchronous I/O
  - **kio**: Handles kernel AIO
  - **pio**: Handles physical logging I/O
  - **lio**: Handles logical logging I/O
  - **gfd**: Global File Descriptor - Each primary and mirror chunk is given a separate global file descriptor. Individual gfd queues are used depending on whether kaio is on and the associated chunk is cooked or raw.

- **len**: The number of pending I/O requests in the queue

- **maxlen**: The largest number of I/O requests that have been in the queue at the same time

- **totalops**: The total number of I/O operations that have been completed for the queue

- **dskread**: Total number of completed read operations for the queue
onstat -g ipl: Print index page logging status information

Syntax:

```
--onstat-- -g--ipl--
```

The onstat -g ipl option shows index page logging status information.

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:20:55 -- 46080 Kbytes
Index page logging status: Enabled
Index page logging was enabled at: 2006/12/20 16:01:02
```

Figure 15-37. onstat -g ipl Output

Output Description:

Index page logging status
Status of index page logging: Enabled or Disabled.

Index page logging was enabled at
The date and time at which index page logging was enabled.

onstat -g iov: Print AIO VP statistics

Syntax:

```
--onstat-- -g--iov--
```

The onstat -g iov option shows asynchronous I/O statistics for each virtual processor.

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 01:56:26 -- 101376 Kbytes
AIO I/O vps:
   class/vp s io/s totalops dskread dskwrite dskcopy wakeups io/wup polltries pollfound kaio_pend
     fifo 0 i 0.0 0 0 0 0 0 0 0
     fifo 1 i 0.0 0 0 0 0 0 0 0
     msc 0 i 0.0 0 0 0 0 0 0 0
     aio 0 s 0.3 628 0 628 0 628 1.0 0 0
```

Figure 15-38. onstat -g iov Output

Output Description:

class
The class of the virtual processor.

vp
The ID number of the virtual processor within its class.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>Current status of the AIO virtual processor</td>
</tr>
<tr>
<td>f</td>
<td>Fork</td>
</tr>
<tr>
<td>i</td>
<td>Idle</td>
</tr>
<tr>
<td>s</td>
<td>Search</td>
</tr>
<tr>
<td>b</td>
<td>Busy</td>
</tr>
<tr>
<td>o</td>
<td>Open</td>
</tr>
<tr>
<td>c</td>
<td>Close</td>
</tr>
<tr>
<td>io/s</td>
<td>The average I/O speed (measured in operations per second) for the virtual processor since the time the database server started or since <code>onstat -z</code> was last run, whichever happened last.</td>
</tr>
<tr>
<td>totalops</td>
<td>Total number of I/O operations performed by this virtual processor since the time the database server started or since <code>onstat -z</code> was last run, whichever happened last.</td>
</tr>
<tr>
<td>dskread</td>
<td>Total number of read operations performed by this virtual processor since the time the database server started or since <code>onstat -z</code> was last run, whichever happened last.</td>
</tr>
<tr>
<td>dskwrite</td>
<td>Total number of write operations performed by this virtual processor since the time the database server started or since <code>onstat -z</code> was last run, whichever happened last.</td>
</tr>
<tr>
<td>dskcopy</td>
<td>Total number of copy operations performed by this virtual processor since the time the database server started or since <code>onstat -z</code> was last run, whichever happened last.</td>
</tr>
<tr>
<td>wakeups</td>
<td>For AIO VPs, the number of times the virtual processor has gone idle since the time the database server started or since <code>onstat -z</code> was last run, whichever happened last.</td>
</tr>
<tr>
<td>io/wup</td>
<td>For AIO VPs, the average number of I/O operations performed per wake-up by this virtual processor since the time the database server started or since <code>onstat -z</code> was last run, whichever happened last.</td>
</tr>
<tr>
<td>polltries</td>
<td>For AIO VPs, the total number of times the kaio thread running on this virtual processor checked the operating system to see if I/O it had requested was done. Count is from the time the database server started or since <code>onstat -z</code> was last run, whichever happened last.</td>
</tr>
<tr>
<td>pollfound</td>
<td>For AIO VPs, the total number of times the kaio thread running on this virtual processor checked the operating system to see if I/O it had requested was done and found that the I/O was complete. Count is from the time the database server started or since <code>onstat -z</code> was last run, whichever happened last.</td>
</tr>
<tr>
<td>kaio_pend</td>
<td>For AIO VPs, the number of I/O requests made by the kaio thread that have not yet completed.</td>
</tr>
</tbody>
</table>

**onstat -g lap**: Print light appends status information

**Syntax:**
The **onstat -g lap** option prints information on the status of light appends occurring in the system.

**Example Output:**

```
Session id (decimal)    Session ID performing the light append operation
address (hexidecimal)  Address of the light append buffer
cur_page (hexidecimal)  Current physical page address
la_npused (decimal)    Number of pages allocated
landata (decimal)      Number of data pages appended
la_nrows (decimal)     Number of rows appended
bufcnt (decimal)       Number of light append buffers
```

**onstat -g lmx: Print all locked mutexes**

**Syntax:**

```
•••onstat -g lmx•••
```

The **onstat -g lmx** option prints all locked mutexes.

**Example Output:**

```
mid   addr   name   holder   lkcnt  waiter  waittime
Locked mutexes: 49
```

**Output Description:**

- **mid**: Internal mutex identifier
- **addr**: Address of locked mutex
- **name**: Name of the mutex
- **holder**: Session ID of the thread holding the mutex
- **lkcnt**: Number of waiters for this mutex
- **waiter**: List of addresses waiting for this mutex
- **waittime**: Amount of time this thread has been waiting
onstat -g lsc: Print active light scan status

Syntax:

```
onstat -g lsc
```

The `onstat -g lsc` option displays the status of any currently active light scans.

Example Output:

```
IBM Informix Dynamic Server Version 11.10.F -- On-Line -- Up 00:08:42 -- 1067288 Kbytes
Light Scan Info
<table>
<thead>
<tr>
<th>descriptor</th>
<th>address</th>
<th>next_lpage</th>
<th>next_ppage</th>
<th>ppage_left</th>
<th>bufcnt</th>
<th>look_aside</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>474b74b0</td>
<td>4a0</td>
<td>7e2c80</td>
<td>416</td>
<td>1</td>
<td>N</td>
</tr>
</tbody>
</table>
```

Figure 15-41. onstat -g lsc Output

Output Description:
- `descriptor (decimal)`: Light scan ID
- `address (hex)`: Memory address of the light scan descriptor
- `next_lpage (hex)`: Next logical page address to scan
- `next_ppage (hex)`: Next physical page address to scan
- `ppage_left (decimal)`: Number of physical pages left to scan in the current extent
- `#bufcnt (decimal)`: Number of light scan buffers used for this light scan
- `#look_aside (char)`: Whether look aside is needed for this light scan (Y = yes, N = no). Look asides occur when a thread needs to examine the buffer pool for existing pages to obtain the latest image of a page being light scanned.

onstat -g mem: Print pool memory statistics

Syntax:

```
onstat -g mem
```

The `onstat -g mem` option prints memory statistics for a pool. Session pools are named with the session number. If no argument is provided, information about all pools is displayed.

Example Output:
Pool Summary:

<table>
<thead>
<tr>
<th>name</th>
<th>class</th>
<th>addr</th>
<th>totalsize</th>
<th>freesize</th>
<th>#allocfrag</th>
<th>#freefrag</th>
</tr>
</thead>
<tbody>
<tr>
<td>resident</td>
<td>R</td>
<td>10a001028</td>
<td>2420736</td>
<td>7960</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>res-buff</td>
<td>R</td>
<td>10a250028</td>
<td>8269824</td>
<td>7960</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>global</td>
<td>V</td>
<td>10aac0028</td>
<td>9351168</td>
<td>32648</td>
<td>650</td>
<td>11</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>onmode_mon</td>
<td>V</td>
<td>10b83028</td>
<td>20480</td>
<td>2752</td>
<td>108</td>
<td>1</td>
</tr>
</tbody>
</table>

Blkpool Summary:

<table>
<thead>
<tr>
<th>name</th>
<th>class</th>
<th>addr</th>
<th>size</th>
<th>#blks</th>
<th>pre-hint</th>
<th>szavail</th>
</tr>
</thead>
<tbody>
<tr>
<td>global</td>
<td>V</td>
<td>10aac8920</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>xmf_msc_pl</td>
<td>V</td>
<td>10ac84ca0</td>
<td>954368</td>
<td>73</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 15-42. onstat -g mem Output

Output Description:

Pool Summary

- **name**: Pool name
- **class**: Shared memory segment type where pool is created
- **addr**: Pool memory address
- **totalsize**: Pool size, in bytes
- **freesize**: Free memory in pool
- **#allocfrag**: Allocated fragments in pool
- **#freefrag**: Free fragments in pool

Blkpool Summary

- **name**: Pool name
- **class**: Shared memory segment type where pool is created
- **addr**: Pool memory address
- **size**: Pool size, in bytes
- **#blks**: Number of blocks in pool

**onstat -g mgm**: Print MGM resource information

**Syntax:**

```
  onstat -g mgm
```

The `onstat -g mgm` option prints Memory Grant Manager (MGM) resource information. You can use the `onstat -g mgm` option to monitor how MGM coordinates memory use and scan threads. This `onstat` option reads shared-memory structures and provides statistics that are accurate at the instant that the command executes.

The `onstat -g mgm` output displays a unit of memory called a *quantum*. The memory *quantum* represents a unit of memory, as follows:
memory quantum = DS_TOTAL_MEMORY / DS_MAX_QUERIES

The following calculation shows the memory quantum for the values that Figure 15-43 displays:
memory quantum = 4000 kilobytes / 31
= 129 kilobytes

The scan thread quantum is always equal to 1.

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:00:51 -- 21504 Kbytes

Memory Grant Manager (MGM)
-------------------------------
MAX_PDQPRIORITY: 100
DS_MAX_QUERIES: 31
DS_MAX_SCANS: 1048576
DS_NONPDQ_QUERY_MEM: 128 KB
DS_TOTAL_MEMORY: 4000 KB

Queries: Active Ready Maximum
0 0 31

Memory: Total Free Quantum
(KB) 4000 4000 128

Scans: Total Free Quantum
1048576 1048576 1

Load Control: (Memory) (Scans) (Priority) (Max Queries) (Reinit)
Gate 1 Gate 2 Gate 3 Gate 4 Gate 5
(Queue Length) 0 0 0 0 0

Active Queries: None
Ready Queries: None

Free Resource Average # Minimum #
Memory 0.0 +- 0.0 500
Scans 0.0 +- 0.0 1048576

Queries Average # Maximum # Total #
Active 0.0 +- 0.0 0 0
Ready 0.0 +- 0.0 0 0

Resource/Lock Cycle Prevention count: 0
```

Figure 15-43. onstat -g mgm Output

Output Description:

The first portion of the output shows the values of the PDQ configuration parameters.

The second portion of the output describes MGM internal control information. It includes four groups of information. The first group is Queries:

**Active** Number of PDQ queries that are currently executing
**Ready**
Number of user queries ready to run but whose execution the database server deferred for load-control reason

**Maximum**
Maximum number of queries that the database server permits to be active. Reflects current value of the DS_MAX_QUERIES configuration parameter

The next group is **Memory**:

**Total**
Kilobytes of memory available for use by PDQ queries (DS_TOTAL_MEMORY specifies this value.)

**Free**
Kilobytes of memory for PDQ queries not currently in use

**Quantum**
Kilobytes of memory in a memory quantum

The next group is **Scans**:

**Total**
The total number of scan threads as specified by the DS_MAX_SCANS configuration parameter

**Free**
Number of scan threads currently available for decision-support queries

**Quantum**
The number of scan threads in a scan-thread quantum

The last group in this portion of the output describes MGM **Load Control**:

**Memory**
Number of queries that are waiting for memory

**Scans**
Number of queries that are waiting for scans

**Priority**
Number of queries that are waiting for queries with higher PDQ priority to run

**Max Queries**
Number of queries that are waiting for a query slot

**Reinit**
Number of queries that are waiting for running queries to complete after an **onmode -M** or **-Q** command

The next portion of the output, **Active Queries**, describes the MGM active and ready queues. This portion of the output shows the number of queries waiting at each gate:

**Session**
The session ID for the session that initiated the query

**Query**
Address of the internal control block associated with the query

**Priority**
PDQ priority assigned to the query

**Thread**
Thread that registered the query with MGM

**Memory**
Memory currently granted to the query or memory reserved for the query (Unit is MGM pages, which is 8 kilobytes.)

**Scans**
Number of scan threads currently used by the query or number of scan threads allocated to the query
**Gate**

Gate number at which query is waiting

The next portion of the output, **Free Resource**, provides statistics for MGM free resources. The numbers in this portion and in the final portion reflect statistics since system initialization or the last onmode -Q, -M, or -S command. This portion of the output contains the following information:

- **Average**: Average amount of memory and number of scans
- **Minimum**: Minimum available memory and number of scans

The next portion of the output, **Queries**, provides statistics concerning MGM queries:

- **Average**: Average active and ready queue length
- **Maximum**: Maximum active and ready queue length
- **Total**: Total active and ready queue length

**Resource/Lock Cycle Prevention count**

Number of times the system immediately activated a query to avoid a potential deadlock. (The database server can detect when some of the queries in its queue might create a deadlock situation if the queries are not run immediately.)

**onstat -g nbm: Print a block bit map**

**Syntax:**

```
>>>onstat -g nbm
```

The **onstat -g nbm** option shows the block bit map for the nonresident segments. Each bit of the bitmap represents a 4 KB block. If the block is used then the bit is set to 1. If it is free the bit is set to 0. The bitmap is shown as a series of hexadecimal numbers. The bits, and therefore the blocks, are numbered starting at 0 so the first block is block 0, the second is block 1, and so on.

**Example Output:**

This example shows the bitmap for the segment of virtual memory at 0x10CC00000. The bitmap itself is at 0x10CC00290. All 1792 blocks of the segment are free except for block 0 and block 1023.

```
Block bitmap for virtual segment address 0x10cc00000:
address = 0x10cc00290, size(bits) = 1792
used = 1, largest_free = -1
  0:8000000000000000 0000000000000000 0000000000000000 0000000000000000
  256:0000000000000000 0000000000000000 0000000000000000 0000000000000000
  512:0000000000000000 0000000000000000 0000000000000000 0000000000000000
  768:0000000000000000 0000000000000000 0000000000000000 0000000000000000
1024:0000000000000000 0000000000000000 0000000000000000 0000000000000000
1280:0000000000000000 0000000000000000 0000000000000000 0000000000000000
1536:0000000000000000 0000000000000000 0000000000000000 0000000000000000
```

**Figure 15-44. onstat -g nbm Output**

**Output Description:**
address  The starting address of the bitmap.
size  The number of bits in the bitmap. This is also the number of 4 KB blocks in the memory segment.
used  The total number of bits in the bitmap that are set to 1. This is also the number of 4 KB blocks that are in use in the memory segment.
largest free  If this is a value other than -1 it is the largest number of consecutive bits that are free, which is also the number of 4 KB blocks in the largest contiguous set of blocks in the memory segment.

A value of -1 means that the largest free space has not been calculated. The database server only calculates the largest free space if it tries to allocate a set of blocks starting at the lastalloc block but there is not enough free space. The value is set to -1 again as soon as another block is allocated in the segment.

**onstat -g nif: Print ER network interface statistics**

**Syntax:**

```plaintext
onstat -g nif [modifier]
```

The `onstat -g nif` command prints statistics about the network interface for Enterprise Replication. The output shows which sites are connected and provides a summary of the number of bytes sent and received by each site. This can help you determine that a site is in a hung state, if it is not sending or receiving bytes.

The `-g nif` option is used primarily as a debugging tool and by Technical Support.

The `onstat -g nif` command has the following formats:

- `onstat -g nif`
- `onstat -g nif modifier`

The following table describes the values for `modifier`.

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>Prints the sum and the sites</td>
</tr>
<tr>
<td>sites</td>
<td>Prints the NIF site context blocks</td>
</tr>
<tr>
<td>serverid</td>
<td>Prints information about the replication server whose groupID is serverID</td>
</tr>
<tr>
<td>sum</td>
<td>Prints the sum of the number of buffers sent and received for each site</td>
</tr>
</tbody>
</table>

**Example Output:**

The following example shows output for the `onstat -g nif` command. In this example, the local server is connected to the server group `g_bombay` and its CDR ID is 200. The connection status is set to running. The server group `g_bombay` NIF version is 7. The local server has sent three messages to the server `g_bombay` and it has received two messages from `g_bombay`. 
onstat -g nsc client_id: Print current shared memory connection information

Syntax:

```bash
onstat -g nsc [client_id]
```

If no `client_id` is provided, information about all current shared memory connections to the database server is given. If a `client_id` is provided then this command gives more detailed information about the shared memory connection with that ID.

Example Output:

This is output of `onstat -g nsc` with no `client_id`. It shows that there is only one user currently connecting to the database server through shared memory. That connection has an ID of 0.

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:02:34 -- 28672 Kbytes
NIF anchor Block: af01610
  nifGState  RUN
  RetryTimeout  300

CDR connections:
  Id  Name  State  Version  Sent  Received
  -----------------------------
  200  g_bombay  RUN  7  3  2
```

Figure 15-45. onstat -g nif Output

This example shows output from running the command using a `client_id` of 0.

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 6 days
  clientid  clientPID  state  #serverbufs  #clientbufs  #rdwrts
  0  6031  Connected  4  4  12
```

Figure 15-46. onstat -g nsc Output
Network Shared Memory Status for Client: 0

<table>
<thead>
<tr>
<th>clientid</th>
<th>clientPID</th>
<th>state</th>
<th>#serverbufs</th>
<th>#clientbufs</th>
<th>#rdwrts</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>18949</td>
<td>Connected</td>
<td>4</td>
<td>4</td>
<td>447048</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>needbuf</th>
<th>segid</th>
<th>semid</th>
<th>semnum</th>
<th>be_semid</th>
<th>be_semnum</th>
<th>be_curread</th>
<th>be_curwrite</th>
<th>fe_curread</th>
<th>fe_curwrite</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1303</td>
<td>851969</td>
<td>0</td>
<td>851969</td>
<td>10</td>
<td>-1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>be_nextread</th>
<th>be_nextwrite</th>
<th>fe_nextread</th>
<th>fe_nextwrite</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>readyqueue</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Server Buffers</th>
<th>Client Buffers</th>
</tr>
</thead>
<tbody>
<tr>
<td>i: bufid</td>
<td>bufid</td>
</tr>
<tr>
<td>status</td>
<td>status</td>
</tr>
<tr>
<td>offset</td>
<td>offset</td>
</tr>
<tr>
<td>fe_addr</td>
<td>fe_addr</td>
</tr>
<tr>
<td>0: 4</td>
<td>0</td>
</tr>
<tr>
<td>inuse 4474</td>
<td>avail 3424</td>
</tr>
<tr>
<td>804474</td>
<td>803424</td>
</tr>
<tr>
<td>1: 5</td>
<td>1</td>
</tr>
<tr>
<td>inuse 4888</td>
<td>avail 3838</td>
</tr>
<tr>
<td>804888</td>
<td>803838</td>
</tr>
<tr>
<td>2: 6</td>
<td>2</td>
</tr>
<tr>
<td>avail 4c9c</td>
<td>avail 3c4c</td>
</tr>
<tr>
<td>8049c</td>
<td>8034c</td>
</tr>
<tr>
<td>3: 7</td>
<td>3</td>
</tr>
<tr>
<td>avail 50b0</td>
<td>avail 4060</td>
</tr>
<tr>
<td>805b0</td>
<td>804060</td>
</tr>
<tr>
<td>4: -1</td>
<td>-1</td>
</tr>
<tr>
<td>free 0</td>
<td>free 0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5: -1</td>
<td>-1</td>
</tr>
<tr>
<td>free 0</td>
<td>free 0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 15-47. onstat -g nsc with client id Output

**Output Description:**

- **clientid**: Server assigned ID
- **clientPID**: Client process ID
- **state**: State of connection
  - *Connected*: The client has established a connection with the server.
  - *Con1*: The server has successfully set up a connection with the client, but the client has not yet been notified of it.
  - *Waiting*: The server is in the process of setting up a connection with the client.
  - *Reject*: Client connection has been rejected by the server, normally because the server is shutting down or not yet in on-line mode.
  - *Closed*: Server has closed the connection with the client. Client might not be aware of the fact yet.
  - *Not connected*: Server is initializing internal structures for the connection.
  - *Unknown*: Connection has been closed and the client is aware of the fact. Server is cleaning up internal structures.
- **#serverbuvs**: Database server buffers currently allocated
- **#clientbuvs**: Client buffers currently allocated
- **#rdwrts**: The total number of reads and writes performed through this connection since it was created.
The following items are only in the output if you run `onstat -g nsd` with a client_id:

- **needbuf**: Indicates if server is waiting for a buffer to be freed
  - 0: False
  - 1: True

- **segid**: Shared memory segment ID
- **semid**: Semaphore ID
- **semnum**: Semaphore number in the semaphore ID
- **be_semid**: Backend semaphore ID
- **be_semnum**: Backend semaphore number in the semaphore ID
- **be_curread**: ID of backend buffer being read
- **be_curwrite**: ID of backend buffer being written
- **fe_curread**: ID of frontend buffer being read
- **fe_curwrite**: ID of frontend buffer being written
- **be_nextread**: ID of next backend buffer to be read
- **be_nextwrite**: ID of next backend buffer to be written
- **fe_nextread**: ID of next frontend buffer to be read
- **fe_nextwrite**: ID of next frontend buffer to be written
- **readyqueue**: Queue of the shared memory buffer ids

**Buffers**

- **i**: Internal location key of message buffer
- **bufid**: Message buffer ID
- **status**: Status of message buffer
- **offset**: Offset of memory buffer in shared memory segments
- **fe_addr**: Frontend address of message buffer

**onstat -g nsd**: Print poll threads shared-memory data

**Syntax:**

```bash
donstat -g nsd
```

The `onstat -g nsd` option prints shared-memory data for poll threads.

**Example Output:**
onstat -g nss: Print shared memory network connections status

Syntax:

```
onstat -g nss [sessionid]
```

The `onstat -g nss` `sessionid` option displays the status of shared memory network connections. If no `sessionid` is provided, a one-line summary for each shared memory connection is listed.

Example Output:

```
clientid  clientPID  state  #serverbufs #clientbufs  #rdwrts
1        14018    Connected    4       4      331
0        12398    Connected    4       4      294
2        14036    Connected    4       4      59
```

Output Description:

- `clientid` (decimal): Server assigned value for lookups
- `clientPID` (decimal): Client process ID
- `state` (string): Current state of the connection.
  - Connected
  - Con1
  - Waiting
  - Reject
  - Badvers
  - Closed
  - Not connected
• Unknown

#serverbufs (dec)
Number of database server buffers currently allocated

#clientbufs (dec)
Number of client buffers currently allocated

#rdwrets (dec)
Total number of buffers in use

onstat -g ntd: Print network statistics

Syntax:

```
  onstat -g ntd
```

The onstat -g ntd option prints network statistics by service.

Example Output:

```

global network information:

<table>
<thead>
<tr>
<th>Client Type</th>
<th>Calls</th>
<th>Accepted</th>
<th>Rejected</th>
<th>Read</th>
<th>Write</th>
<th>q-free</th>
<th>q-limits</th>
<th>q-exceed</th>
<th>alloc/max</th>
</tr>
</thead>
<tbody>
<tr>
<td>sqlexec</td>
<td>yes</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1/1</td>
<td>135/10</td>
<td>0/0</td>
</tr>
<tr>
<td>srvinfx</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>onspace</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>onlog</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>onparam</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>oncheck</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>onmonitor</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dr_accept</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ontape</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>srvstat</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>asfecho</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>listener</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>crsamexec</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>onutil</td>
<td>yes</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Totals</td>
<td></td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 15-50. onstat -g ntd Output

onstat -g ntm: Print network mail statistics

Syntax:

```
  onstat -g ntm
```

The onstat -g ntm option prints network mail statistics.

Example Output:
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 01:57:33 -- 101376 Kbytes

global network information:

<table>
<thead>
<tr>
<th>netscb</th>
<th>connects</th>
<th>read</th>
<th>write</th>
<th>q-free</th>
<th>q-limits</th>
<th>q-exceed</th>
<th>alloc/max</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1/ 1</td>
<td>135/ 10</td>
<td>0/ 0</td>
</tr>
</tbody>
</table>

Network mailbox information:

<table>
<thead>
<tr>
<th>box</th>
<th>netscb</th>
<th>thread name</th>
<th>max received</th>
<th>in box</th>
<th>max in box</th>
<th>full signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>c631028</td>
<td>tlitcppoll</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>c63e548</td>
<td>tlitclplst</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>no</td>
</tr>
</tbody>
</table>

Figure 15-51. onstat -g ntm Output

**onstat -g ntt: Print network user times**

Syntax:

```
$ onstat -g ntt
```

The `onstat -g ntt` option prints network user times.

**Example Output:**

IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 01:57:44 -- 101376 Kbytes
global network information:

<table>
<thead>
<tr>
<th>netscb</th>
<th>connects</th>
<th>read</th>
<th>write</th>
<th>q-free</th>
<th>q-limits</th>
<th>q-exceed</th>
<th>alloc/max</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1/ 1</td>
<td>135/ 10</td>
<td>0/ 0</td>
</tr>
</tbody>
</table>

Individual thread network information (times):

<table>
<thead>
<tr>
<th>netscb</th>
<th>thread name</th>
<th>sid</th>
<th>open</th>
<th>read</th>
<th>write</th>
<th>address</th>
</tr>
</thead>
<tbody>
<tr>
<td>c76ea28</td>
<td>ontape</td>
<td>61</td>
<td>14:34:48</td>
<td>14:34:50</td>
<td>14:34:50</td>
<td></td>
</tr>
<tr>
<td>c63e548</td>
<td>tlitclplst</td>
<td>4</td>
<td>14:30:43</td>
<td>14:34:48</td>
<td>server.ibm.com</td>
<td>5006</td>
</tr>
<tr>
<td>c631028</td>
<td>tlitcppoll</td>
<td>3</td>
<td>14:32:32</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 15-52. onstat -g ntt output

**onstat -g ntu: Print network user statistics**

Syntax:

```
$ onstat -g ntu
```

The `onstat -g ntu` option prints network user statistics.

**Example Output:**
onstat -g opn: Print open partitions

Syntax:

```
onstat -g opn [thread ID]
```

The `onstat -g opn` option prints a list of the partitions (tables/indexes), by thread ID, that are currently open in the system. The `onstat -g opn thread ID` option restricts the list to the specified ID.

Example Output:
Figure 15-54. onstat -g opn Output

Output Description:

<table>
<thead>
<tr>
<th>tid (decimal)</th>
<th>Thread ID currently accessing the partition resource (table/index)</th>
</tr>
</thead>
<tbody>
<tr>
<td>rstcb (hex)</td>
<td>In-memory address of the RSAM thread control block for this thread</td>
</tr>
<tr>
<td>isfd (decimal)</td>
<td>ISAM file descriptor associated with the open partition</td>
</tr>
<tr>
<td>op_mode (hex)</td>
<td>Current status of the partition lock mode using a combination of the following hexadecimal values:</td>
</tr>
<tr>
<td></td>
<td>0x000000 Open for input only</td>
</tr>
<tr>
<td></td>
<td>0x000001 Open for output only</td>
</tr>
<tr>
<td></td>
<td>0x000002 Open for input and output</td>
</tr>
<tr>
<td></td>
<td>0x000004 System catalog</td>
</tr>
<tr>
<td></td>
<td>0x000008 No logical logging</td>
</tr>
<tr>
<td></td>
<td>0x000010 Open if not already opened for alter</td>
</tr>
<tr>
<td></td>
<td>0x000020 Open all fragments data and index</td>
</tr>
<tr>
<td></td>
<td>0x000040 Do not allocate a blob descriptor</td>
</tr>
<tr>
<td></td>
<td>0x000080 Open for alter</td>
</tr>
<tr>
<td></td>
<td>0x000100 Open all data fragments</td>
</tr>
<tr>
<td></td>
<td>0x000200 Automatic record lock</td>
</tr>
<tr>
<td></td>
<td>0x000400 Manual record lock</td>
</tr>
<tr>
<td></td>
<td>0x000800 Exclusive ISAM file lock</td>
</tr>
</tbody>
</table>
0x001000 Ignore dataskip - data cannot be ignored
0x002000 Dropping partition - delay file open
0x004000 Do not drop blobspace blobs when table dropped
    (alter fragment)
0x010000 Open table for DDL operations
0x040000 Do not assert fail if this partnum does not exist
0x080000 Include fragments of subtables
0x100000 Table created under supertable
0x400000 Blob in use by CDR

op_flags (hex)   Current status of the partition using a combination of the following
                 hexadecimal values:
     0x0001  Open data structure is in use
     0x0002  Current position exists
     0x0004  Current record has been read
     0x0008  Duplicate created or read
     0x0010  Skip current record on reverse read
     0x0020  Shared blob information
     0x0040  Partition opened for rollback
     0x0080  Stop key has been set
     0x0100  No index related read aheads
     0x0200  isstart called for current stop key
     0x0400  Pseudo-closed
     0x0800  Real partition opened for SMI query
     0x1000  Read ahead of parent node is done
     0x2000  UDR keys loaded
     0x4000  Open is for a pseudo table
     0x8000  End of file encountered when positioning in table

partnum (hex)   Partition number for the open resource (table/index)
ucount (decimal) Number of user threads currently accessing this partition
ocount (decimal) Number of times this partition was opened
lockmode (decimal) Type of lock being held using one of the following coded values:
    0   No locks
    1   Byte lock
    2   Intent shared lock
    3   Shared lock
    4   Shared lock by repeatable read (only on items)
    5   Update lock
    6   Update lock by repeatable read (only on items)
    7   Intent exclusive lock
    8   Shared, intent exclusive lock
    9   Exclusive lock
   10   Exclusive lock by repeatable read (only on items)
   11   Inserter's repeatable read test lock

onstat -g pos: Print file values

Syntax:

    onstat -g pos

The onstat -g pos option prints the values for the $INFORMIXDIR/etc/.
.infos.DBSERVERNAME file.

Example Output:

---
onstat -g ppf partition number | 0: Print partition profiles

Syntax:

```bash
onstat -g ppf [partition number | 0]
```

The `onstat -g ppf` option prints the partition profile for `partition number`; 0 prints profiles for all partitions. If `TBLSPACE_STATS` configuration parameter is set to 0, displays: Partition profiles disabled.

Example Output:

<table>
<thead>
<tr>
<th>Partition profiles</th>
<th>partnum</th>
<th>lkrqs</th>
<th>lkwts</th>
<th>dlks</th>
<th>touts</th>
<th>isrd</th>
<th>iswrt</th>
<th>isrwt</th>
<th>isdel</th>
<th>bfrd</th>
<th>bfwrt</th>
<th>seqsc</th>
<th>rhitratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x1000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0x100002</td>
<td>1506</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>416</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td>1282</td>
<td>20</td>
<td>0</td>
<td>97</td>
</tr>
<tr>
<td>0x100003</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>75</td>
</tr>
<tr>
<td>0x1000a5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>67</td>
</tr>
<tr>
<td>0x1000e3</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>0x2000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0x3000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0x4000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Output Description:

- `partnum`: Partition number
- `lkrqs`: Lock requests
- `lkwts`: Lock waits
- `dlks`: Deadlocks

Figure 15-55. onstat -g pos Output

Figure 15-56. onstat -g ppf Output

Chapter 15. The onstat Utility 15-75
onstat -g prc: Print sessions using UDR or SPL routine

Syntax:

```
onstat -g prc
```

The `onstat -g prc` option prints the number of sessions currently using the UDR or SPL routine.

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 01:59:00 -- 101376 Kbytes
Stored Procedure Cache:
  Number of lists : 31
  PC_POOLSIZE   : 50
  Number of entries : 0
  Number of inuse entries : 0
Stored Procedure Cache Entries:
  list#  id  ref_cnt  dropped?  heap_ptr  procedure name
  --------------------------------------------------------------
  Stored Procedure Cache is empty.
```

Figure 15-57. onstat -g prc Output

onstat -g que: Prints ER queue statistics

Syntax:

```
onstat -g que
```

The `onstat -g que` command prints statistics that are common to all queues in Enterprise Replication. The queuer manages the logical aspects of the queue. The RQM (reliable queue manager) manages the physical queue.

The `-g que` option is used primarily as a debugging tool and by Technical Support.

Example Output:

In the following example, **Element high water mark** shows the maximum size of the transaction buffer header data (metadata) allowed in memory, shown in kilobytes. **Data high water mark** shows the maximum size of transactions for user
data allowed in memory, shown in kilobytes.

IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:40:28 -- 28672 Kbytes
CDR Queueer Statistics:
  Queuer state          : 2
  Local server          : 100
  Element high water mark : 131072
  Data high water mark  : 131072
  # of times txns split : 0
  Total # of split txns : 0
  allowed log delta     : 30
  maximum delta detected: 4
  Control Key           : 0/00000007
  Synchronization Key   : 0/00000003
Replay Table:
  Replay Posn (Disk value): 12/000000018 (12/000000018)
  Replay save interval   : 10
  Replay updates         : 10
  Replay # saves         : 17
  Replay last save time  : (1095118157) 2004/09/13 18:29:17
Send Handles
  Server ID            : 200
  Send state,count     : 0,0
  RQM hdl for trg_send: Traverse handle (0xaf8e018) for thread CDRACK_0 at Head_of_Q,
                        Flags: None
  RQM hdl for control_send: Traverse handle (0xaf74018)
                           for thread CDRACK_0 at Head_of_Q, Flags: None
  RQM hdl for sync_send: Traverse handle (0xadc6018) for thread CDRACK_0 at Head_of_Q,
                         Flags: None
  Server ID            : 200
  Send state,count     : 0,0
  RQM hdl for trg_send: Traverse handle (0xac8b018) for thread CDRACK_1 at Head_of_Q,
                        Flags: None
  RQM hdl for control_send: Traverse handle (0xb1ce018)
                           for thread CDRACK_1 at Head_of_Q, Flags: None
  RQM hdl for sync_send: Traverse handle (0xadc5018) for thread CDRACK_1 at Head_of_Q,
                         Flags: None
  Server ID            : 200
  Send state,count     : 0,0
  RQM hdl for trg_send: Traverse handle (0xaea71d8) for thread CDRNsA200 at Head_of_Q,
                        Flags: None
  RQM hdl for ack_send: Traverse handle (0xae8c1d8) for thread CDRNsA200 at Head_of_Q,
                        Flags: None
  RQM hdl for control_send: Traverse handle (0xae9e1d8) for thread CDRNsA200 at Head_of_Q,
                         Flags: None

Figure 15-58. onstat -g que Output

**onstat -g qst: Print wait options for mutex and condition queues**

**Syntax:**

```
—onstat —g—qst
```

The **onstat -g qst** option displays wait statistics for mutex queues and condition queues (queues of waiters for a mutex or a condition). The QSTATS configuration parameter must be set to 1 to enable statistics collection. For more information, see "QSTATS" on page 1-77

**Example Output:**

Mutex Queue Statistics
name   nwaits avg_time max_time avgq maxq nservs avg_time

  ddh  chai 1     13354863 13354863  1   1   56  1690

Condition Queue Statistics
name   nwaits avg_time max_time avgq maxq nservs avg_time

  arrived  1    110008  110008   1   1   0    0
  logbf0  21     642    4431   1   2   0    0
  logbf1  15     475    2519   1   2   0    0
  logbf2  19     596    3274   1   2   0    0
  bp_cond 1     0    0   1   1   0    0

Figure 15-59. `onstat -g qst` Output

Output Description:
- **name (string)**: Name of the mutex or condition resource being waited for
- **nwaits (decimal)**: Number of times this resource was waited for
- **avg_time (decimal)**: Average time spent waiting (in microseconds)
- **max_time (decimal)**: Maximum time spent waiting (in microseconds)
- **avgq (decimal)**: Average length of the queue
- **maxq (decimal)**: Maximum length of the queue
- **nservs (decimal)**: Number of times this resource was acquired
- **avg_time (decimal, microsecond)**: Average time the resource was held per acquisition (in microseconds)

**onstat -g rbm**: Print a block map of shared memory

Syntax:

```plaintext
>>> onstat -g rbm ...
```

The `onstat -g rbm` option prints a block map for the resident segment of shared memory.

Example Output:
### Output Description:

#### Header
- **address (hex)**: In-memory starting address of the used/free blocks in the segment.
- **size (bits)**: Number of bits in the block bitmap; each bit represents one block.
- **used (blocks)**: Used blocks in the bitmap.
- **largest_free (blocks)**: Largest run of free blocks.

#### Data
- **Bit number (decimal): data (hex)**: Bit number followed by 32 bytes of data (hex).

### onstat -g rcv: Print ER receive manager statistics

#### Syntax:

```bash
>>> onstat -g rcv [serverid full]
```

The `onstat -g rcv` command prints statistics about the receive manager in Enterprise Replication. The receive manager is a set of service routines between the receive queues and data sync.

The `onstat -g rcv` command has the following formats:
- `onstat -g rcv`
- `onstat -g rcv serverid`
- `onstat -g rcv full`

The `serverID` modifier causes the command to print only those output messages received from the replication server whose groupID is `serverid`. The `full` modifier causes the command to print all statistics.

The `onstat -g rcv` command includes the Receive Manager global section. In this section, the following fields have the meanings shown:

---

**Figure 15-60. onstat -g rbm Output**

Block bitmap for resident segment address 0x44000000:

address = 0x440003bc, size(bits) = 3035
used = 3031, largest_free = 4

<table>
<thead>
<tr>
<th>Block Size</th>
<th>Bitmap</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>256</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>512</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>768</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>1024</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>1280</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>1536</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>1792</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>2048</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>2304</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>2560</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>2816</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>2048</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>2304</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>2560</td>
<td>00000000:ffff00</td>
</tr>
<tr>
<td>2816</td>
<td>00000000:ffff00</td>
</tr>
</tbody>
</table>

---

Chapter 15. The onstat Utility 15-79
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdrRM_DSParallelPL</td>
<td>Shows the current level of Apply Parallelism, 0 (zero) being the highest</td>
</tr>
<tr>
<td>cdrRM_DSNumLockTimeout</td>
<td>Indicate the number of collisions between various apply threads</td>
</tr>
<tr>
<td>cdrRM_DSNumLockRB</td>
<td></td>
</tr>
<tr>
<td>cdrRM_DSNumDeadLocks</td>
<td></td>
</tr>
<tr>
<td>cdrRM_acksinList</td>
<td>Shows acknowledgements that have been received but not yet processed</td>
</tr>
</tbody>
</table>

The `onstat -g rcv` command includes the Receive Parallelism Statistics section, a summary of the data sync threads by source server.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server</td>
<td>Source server ID</td>
</tr>
<tr>
<td>Tot.Txn.</td>
<td>Total number of transactions applied from this source server</td>
</tr>
<tr>
<td>Pending</td>
<td>Number of current transactions in the pending list for this source server</td>
</tr>
<tr>
<td>Active</td>
<td>Number of current transactions currently being applied from this source server</td>
</tr>
<tr>
<td>MaxPnd</td>
<td>Maximum number of transactions in the pending list queue</td>
</tr>
<tr>
<td>MaxAct</td>
<td>Maximum number of transaction in the active list queue</td>
</tr>
<tr>
<td>AvgPnd</td>
<td>Average depth of the pending list queue</td>
</tr>
<tr>
<td>AvgAct</td>
<td>Average depth of the active list queue</td>
</tr>
<tr>
<td>CommitRt</td>
<td>Commit rate of transaction from this source server based on transactions per second</td>
</tr>
</tbody>
</table>

The Statistics by Source section of the `onstat -g rcv` command shows the following information for each source server. For each replicate ID:

- The number of transactions applied from the source servers
- The number of inserts, deletes, and updates within the applied transactions
- The timestamp of the most recently applied transaction on the target server
- The timestamp of the commit on the source server for the most recently applied transaction

The `-g rcv` option is used primarily as a debugging tool and by Technical Support. If you suspect that acknowledgement messages are not being applied, you can use this option to check.

**Example Output:**

The following example shows output for the `onstat -g rcv full` command.
Receive Manager global block 0D452018

- cdrRM_inst_ct: 5
- cdrRM_State: 00000000
- cdrRM_numSleepers: 3
- cdrRM_DsCreated: 3
- cdrRM_MinDSThreads: 1
- cdrRM_MaxDSThreads: 4
- cdrRM_DSBlock: 0
- cdrRM_DSParallelPL: 0
- cdrRM_DSFailRate: 0.000000
- cdrRM_DSNumRun: 35
- cdrRM_DSNumLockTimeout: 0
- cdrRM_DSNumLockRB: 0
- cdrRM_DSNumDeadLocks: 0
- cdrRM_DSNumPCommits: 0
- cdrRM_ACKwaiting: 0
- cdrRM_totSleep: 77
- cdrRM_Sleeptime: 153
- cdrRM_Workload: 0
- cdrRM_optscale: 4
- cdrRM_MinFloatThreads: 2
- cdrRM_FloatThreadCount: 2
- cdrRM_ACKWaiting: 2
- cdrRM_AckCreateStamp: Wed Sep 08 11:47:49 2004
- cdrRM_DSCreateStamp: Wed Sep 08 14:16:35 2004
- cdrRM_acksInList: 0
- cdrRM_BlobErrorBufs: 0

Figure 15-61. onstat -g rcv Output (Part 1 of 2)

Receive Parallelism Statistics

<table>
<thead>
<tr>
<th>Srvr</th>
<th>Tot.Txn</th>
<th>Pnding</th>
<th>Active</th>
<th>MaxPnd</th>
<th>MaxAct</th>
<th>AvgPnd</th>
<th>AvgAct</th>
<th>CommitRt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>35</td>
<td>0</td>
<td>0</td>
<td>21</td>
<td>3</td>
<td>7.00</td>
<td>1.63</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
<td>0.02</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1.00</td>
<td>1.00</td>
<td>0.21</td>
</tr>
</tbody>
</table>

Tot Pending:0 Tot Active:0 Avg Pending:5.77 Avg Active:1.50
Commit Rate:0.01

Time Spent In RM Parallel Pipeline Levels

<table>
<thead>
<tr>
<th>Lev.</th>
<th>TimeInSec</th>
<th>Pcnt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>17405</td>
<td>100.00%</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.00%</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0.00%</td>
</tr>
</tbody>
</table>

Statistics by Source

Server 1

<table>
<thead>
<tr>
<th>Repl</th>
<th>Txn</th>
<th>Ins</th>
<th>Del</th>
<th>Upd</th>
<th>Last</th>
<th>Target</th>
<th>Apply</th>
<th>Last</th>
<th>Source</th>
<th>Commit</th>
</tr>
</thead>
<tbody>
<tr>
<td>65541</td>
<td>23</td>
<td>0</td>
<td>1</td>
<td>616</td>
<td>2004/09/08 14:20:15</td>
<td>2004/09/08 14:20:15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>65542</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>253</td>
<td>2004/09/08 14:19:33</td>
<td>2004/09/08 14:19:33</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>65545</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>67</td>
<td>2004/09/08 14:20:37</td>
<td>2004/09/08 14:20:37</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Server 5

<table>
<thead>
<tr>
<th>Repl</th>
<th>Txn</th>
<th>Ins</th>
<th>Del</th>
<th>Upd</th>
<th>Last</th>
<th>Target</th>
<th>Apply</th>
<th>Last</th>
<th>Source</th>
<th>Commit</th>
</tr>
</thead>
<tbody>
<tr>
<td>65541</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>81</td>
<td>2004/09/08 16:36:10</td>
<td>2004/09/08 16:36:09</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Server 6

<table>
<thead>
<tr>
<th>Repl</th>
<th>Txn</th>
<th>Ins</th>
<th>Del</th>
<th>Upd</th>
<th>Last</th>
<th>Target</th>
<th>Apply</th>
<th>Last</th>
<th>Source</th>
<th>Commit</th>
</tr>
</thead>
<tbody>
<tr>
<td>65548</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>42</td>
<td>2004/09/08 16:37:59</td>
<td>2004/09/08 16:37:58</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 15-61. onstat -g rcv Output (Part 2 of 2)
onstat -g rea: Print ready threads

Syntax:

```bash
onstat -g rea
```

The `onstat -g rea` option prints ready threads.

Example Output:

```
Figure 15-62. onstat -g rea Output
```

onstat -g rep: Print ER schedule manager events

Syntax:

```bash
onstat -g rep replname
```

The `onstat -g rep` command prints events that are in the queue for the schedule manager for Enterprise Replication. The `-g rep` option is used primarily as a debugging tool and by Technical Support.

The `onstat -g rep` command has the following formats:

- `onstat -g rep`
- `onstat -g rep replname`

The `repl_name` modifier limits the output to those events originated by the replicate named `repl_name`.

Example Output:

```
Figure 15-63. onstat -g rep Output
```

onstat -g rqm: Print low-level queue statistics

Syntax:

```bash
```
The **onstat -g rqm** command prints statistics and contents of the low-level queues (send queue, receive queue, ack send queue, sync send queue, and control send queue) managed by the Reliable Queue Manager (RQM) in Enterprise Replication. The RQM manages the insertion and removal of items to and from the various queues. The RQM also manages spooling of the in-memory portions of the queue to and from disk. The **-g rqm** option displays the contents of the queue, size of the transactions in the queue, how much of the queue is in memory and on disk, the location of various handles to the queue, and the contents of the various progress tables. You can choose to print information for all queues or for just one queue by using one of the modifiers described below.

If a queue is empty, no information is printed for that queue.

The **onstat -g rqm** command has the following formats:

```
onstat -g rqm  
onstat -g rqm modifier
```

The following table describes the values for `modifier`.

<table>
<thead>
<tr>
<th>Modifier</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKQ</td>
<td>Prints the ack send queue</td>
</tr>
<tr>
<td>BRIEF</td>
<td>Prints a brief summary of the number of transactions in each of the queues and the replication servers for which the data is queued. Use this modifier to quickly identify sites where a problem exists. If large amounts of data are queued for a single server, then that server is probably down or off the network.</td>
</tr>
<tr>
<td>CNTRLQ</td>
<td>Prints the control send queue</td>
</tr>
<tr>
<td>FULL</td>
<td>Prints full information about every in-memory transaction for every queue</td>
</tr>
<tr>
<td>RECVQ</td>
<td>Prints the receive queue</td>
</tr>
<tr>
<td>SBSPACES</td>
<td>Prints information about the sbspaces configured for ER</td>
</tr>
<tr>
<td>SENDQ</td>
<td>Prints the send queue</td>
</tr>
<tr>
<td>SYNCQ</td>
<td>Prints the sync send queue</td>
</tr>
<tr>
<td>VERBOSE</td>
<td>Prints all the buffer headers in memory</td>
</tr>
</tbody>
</table>

When you specify a modifier to select a specific queue, the command prints all the statistics for that queue and information about the first and last in-memory transactions for that queue.

The other modifiers of the **onstat -g rqm** command are used primarily as a debugging tool and by Technical Support.

The output for the SENDQ modifier contains the following sections:

- RQM Statistics for Queue—a summary of current and historical information for the queue. This includes the number of transactions in the queue, how many are spooled, how many bytes they are using, some maximum statistics, and the high water marks that will trigger stably storing transactions in the `syscdr` tables.
- First Txn—information about the first transaction in the queue. To check if the queue is draining, you can run **onstat -g rqm** several times and see if the first
transaction’s RQM key is changing. The RQM key has the following format:
Server_ID/Commit_unique_logID/Commit_log_position/Sequence. If it is not
draining, the target server may be offline or some other problem is occurring.
The NeedAck field shows from which server the transaction is waiting for an
acknowledgement. You can use this bitmap mask with the output from the
onstat -g cat command to determine the name of the server which server
Enterprise Replication is waiting on for an acknowledgement.

- Last Txn—information about the last transaction in the queue
- Traverse handle—lists the handles used for threads
- Progress table—provides information about the progress of each replicate under
  the headers: Server, Group, Bytes Queued, Acked, and Sent. The Group field
  shows the replicate ID. The Acked field shows what has been acknowledged.
  The Sent field shows which entries are now in transit. Both the Acked and the
  Sent field show the RQM key, which has the following format:
  Server_ID/Commit_unique_logID/Commit_log_position/Sequence.

Example Output:

The following example shows output for the onstat -g rqm SENDQ command.

```
RQM Statistics for Queue (0x003DF018) trg_send
Transaction Spool Name: trg_send_stxn
Insert Stamp: 35/0
Flags: SEND_Q, SPOOLED, PROGRESS_TABLE, NEED_ACK
Txns in queue: 35
Log Events in queue: 0
Txns in memory: 35
Txns in spool only: 0
Txns spooled: 0
Unspooled bytes: 176206
Size of Data in queue: 176206 Bytes
Real memory in use: 176206 Bytes
Pending Txn Buffers: 0
Pending Txn Data: 0 Bytes
Max Real memory data used: 176206 (2457600) Bytes
Max Real memory hds used: 65988 (2457600) Bytes
Total data queued: 176206 Bytes
Total Txns queued: 35
Total Txns spooled: 0
Total Txns restored: 0
Total Txns recovered: 0
Spool Rows read: 0
Total Txns deleted: 0
Total Txns duplicated: 0
Total Txn Lookups: 363
```

Figure 15-64. onstat -g rqm SENDQ Output (Part 1 of 3)
onstat -g rss: Print RS secondary server information

Syntax:

```
--onstat -g rss [verbose log server_name]
```
The `onstat -g rss` command prints RS secondary server information. The output of the `onstat -g rss` command differs slightly depending on whether the command is run on the primary server or on the RS secondary server.

<table>
<thead>
<tr>
<th>Invocation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>onstat -g rss</code></td>
<td>Displays brief RS secondary server information</td>
</tr>
<tr>
<td><code>onstat -g rss verbose</code></td>
<td>Displays detailed RS secondary server information</td>
</tr>
<tr>
<td><code>onstat -g rss log</code></td>
<td>Displays log information</td>
</tr>
<tr>
<td><code>onstat -g rss server_name</code></td>
<td>Displays information about a specific RS secondary server</td>
</tr>
</tbody>
</table>

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1    -- On-Line -- Up 00:08:56 -- 47104 Kbytes

Local server type: Primary
Index page logging status: Enabled
Index page logging was enabled at: 2007/02/20 18:10:01
Number of RSS servers: 3

RSS Server information:

<table>
<thead>
<tr>
<th>RSS Srv name</th>
<th>RSS Srv name</th>
<th>Connection status</th>
<th>Next LPG to send</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdr_ol_nag_1_c1</td>
<td>Active</td>
<td>Connected</td>
<td>7,899</td>
</tr>
<tr>
<td>cdr_ol_nag_1_c2</td>
<td>Active</td>
<td>Connected</td>
<td>7,899</td>
</tr>
<tr>
<td>cdr_ol_nag_1_c3</td>
<td>Active</td>
<td>Connected</td>
<td>7,899</td>
</tr>
</tbody>
</table>

Figure 15-65. `onstat -g rss` Output (run on primary server)
```

Output Description:

Local server type
- Primary or RSS (remote standalone secondary) server type

Index page logging status
- Displays whether index page logging is enabled or disabled between primary server and secondary server

Index page logging was enabled at
- Date and time that index page logging was enabled

Number of RSS servers
- Number of RS secondary servers connected to the primary server

RSS Srv name
- Name of RS secondary server

RSS Srv status
- Displays whether RS secondary server is active

Connection status
- Connection status of RS secondary server

Next LPG to send (log id, page)
- LPG log ID and page

Example Output:
IBM Informix Dynamic Server Version 11.10.UC1 -- Read-Only (RSS) -- Up 00:05:18 -- 55296 Kbytes

Local server type: RSS
Server Status : Active
Source server name: cdr_ol_nag_1
Connection status: Connected
Last log page received (log id,page): 7,877

Figure 15-66. onstat -g rss Output (run on RS secondary server)

Output Description:

Local server type
Primary or RSS (remote standalone secondary) server type

Server Status
Displays whether RS secondary server is active

Source server name
Name of the primary server

Connection status
Connection status of RS secondary server

Last log page received (log id,page)
Most recent log ID and page received

Example Output:

IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:08:57 -- 47104 Kbytes

Log Pages Snooped:
<table>
<thead>
<tr>
<th>RSS Srv name</th>
<th>From Cache</th>
<th>From Disk</th>
<th>Tossed (LBC full)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdr_ol_nag_1_c1</td>
<td>1368</td>
<td>1331</td>
<td>0</td>
</tr>
<tr>
<td>cdr_ol_nag_1_c2</td>
<td>1357</td>
<td>1342</td>
<td>0</td>
</tr>
<tr>
<td>cdr_ol_nag_1_c3</td>
<td>1356</td>
<td>1343</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 15-67. onstat -g rss log Output (run on primary server)

Output Description:

Log Pages Snooped
Statistics for each RS secondary server

RSS Srv name
RS secondary server name

From Cache
From cache number

From Disk
Log from disk

Tossed (LBC full)
Number of log pages discarded as a result of the LBC becoming full
**onstat -g rwm: Print read and write mutexes**

**Syntax:**

```
  onstat -g rwm
```

The `onstat -g rwm` command prints read and write mutexes.

**Example Output:**

```
MUTEX_NAME  write/read/wait  tcb list
<address> <name> first mutex
  Writer  ticket = <ticket address>  tcb=<thread address> <thread name>
  Readers ticket = <ticket address>  tcb=<thread address> <thread name>
  Waiters ticket = <ticket address>  tcb=<thread address> <thread name>
<address> <name> second mutex
  Writer  ticket = <ticket address>  tcb=<thread address> <thread name>
  Readers ticket = <ticket address>  tcb=<thread address> <thread name>
  Waiters ticket = <ticket address>  tcb=<thread address> <thread name>
  ....
  ....
<address> <name> last mutex
  Writer  ticket = <ticket address>  tcb=<thread address> <thread name>
  Readers ticket = <ticket address>  tcb=<thread address> <thread name>
  Waiters ticket = <ticket address>  tcb=<thread address> <thread name>
```

*Figure 15-68. onstat -g rwm Output*

**Output Description:**

- **tcb** List of thread addresses
- **Writer** List of write threads
- **Readers** List of read threads
- **Waiters** List of waiting threads
- **ticket** Address of ticket acquired by the thread

**onstat -g sch: Print VP information**

**Syntax:**

```
  onstat -g sch
```

The `onstat -g sch` option prints the number of semaphore operations, spins, and busy waits for each virtual processor.

**Example Output:**
onstat -g sds: Print SDS secondary server information

Syntax:

```
----onstat- -g-sds---
                  [verbose-server_name]
```

The `onstat -g sds` command prints SDS secondary server information. The output of the `onstat -g sds` command differs slightly depending on whether the command is run on the primary server or on the SD secondary server.

<table>
<thead>
<tr>
<th>Invocation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>onstat -g sds</td>
<td>Displays brief SD secondary server information</td>
</tr>
<tr>
<td>onstat -g sds verbose</td>
<td>Displays detailed SD secondary server information</td>
</tr>
<tr>
<td>onstat -g sds server_name</td>
<td>Displays information about a specific SD secondary server. When server_name is specified, the command must be issued from the primary server.</td>
</tr>
</tbody>
</table>

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 02:00:03 -- 101376 Kbytes
VP Scheduler Statistics:
  vp pid  class  semops  busy waits  spins/wait
 1  2599  cpu     0       0          0
 2  2602  adm     0       0          0
 3  2603  cpu  125735  125735     10001
 4  2604  cpu  125486  125487     10291
 5  2605  cpu  125585  125585     10001
 6  2606  lio  811      811       1000 
 7  2607  pio  810      810       1000
 8  2608  aio  1489    1489       1000
 9  2609  msc  810      810       1000
10 2610  fifo  810     810       1000
11 2611  fifo  811     811       1000
12 2612  aio  812      812       1000
13 2613  aio  810      810       1000
14 2614  aio  811      811       1000
Thread Migration Statistics:
  vp pid  class  steal-at  steal-sc  idlvp-at  idlevp-sc  Q-Ln  Polls  Idles  IdleSec
 1  2599  cpu  25799     184       0        0    0           0          0  0
 2  2602  adm  0         0        0        0    0           0          0  0
 3  2603  cpu  125596    43        1       0     0           0          0  0
 4  2604  cpu  125481    8        2       0     0           0          0  0
 5  2605  cpu  125594   17       0       0     0           0          0  0
 6  2606  lio  0         0        0       0    0           0          0  0
 7  2607  pio  0         0        0       0    0           0          0  0
 8  2608  aio  0         0        0       0    0           0          0  0
 9  2609  msc  0         0        0       0    0           0          0  0
10 2610  fifo  0         0        0       0    0           0          0  0
11 2611  fifo  0         0        0       0    0           0          0  0
12 2612  aio  0         0        0       0    0           0          0  0
13 2613  aio  0         0        0       0    0           0          0  0
14 2614  aio  0         0        0       0    0           0          0  0
```

Figure 15-69. onstat -g sch Output
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:06:17 -- 38912 Kbytes

Local server type: Primary
Number of SDS servers: 2

SDS server information

SDS srv SDS srv Connection Last LPG sent
name status tatus (log id, page)
cdr_ol_nag_1_sdc1 Active Connected 7,884
cdr_ol_nag_1_sdc2 Active Connected 7,884

Figure 15-70. onstat -g sds Output (run from primary server)

Output Description:

Local server type
   Primary or SDS (shared disk secondary) server type

Number of SDS servers
   Number of SD secondary servers connected to the primary server

SDS Srv name
   Name of SD secondary server

SDS Srv status
   Displays whether SD secondary server is active

Connection status
   Displays whether SD secondary server is connected

Last LPG sent (log id, page)
   Most recent LPG log ID and page

Example Output:
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:06:17 -- 38912 Kbytes

<table>
<thead>
<tr>
<th>Number of SDS servers: 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Updater node alias name: cdr_ol_nag_1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SDS server control block: 0xb3cd8d0</th>
</tr>
</thead>
<tbody>
<tr>
<td>server name: cdr_ol_nag_1_sdc1</td>
</tr>
<tr>
<td>server type: SDS</td>
</tr>
<tr>
<td>server status: Active</td>
</tr>
<tr>
<td>connection status: Connected</td>
</tr>
<tr>
<td>Last log page sent (log id, page): 7,884</td>
</tr>
<tr>
<td>Last log page flushed (log id, page): 7,884</td>
</tr>
<tr>
<td>Last LSN acked (log id, pos): 7,3621272</td>
</tr>
<tr>
<td>Sequence number of next buffer to send: 176</td>
</tr>
<tr>
<td>Sequence number of last buffer acked: 0</td>
</tr>
<tr>
<td>Time of last ack: 2007/02/20 21:04:13</td>
</tr>
<tr>
<td>Total LSNs posted: 0</td>
</tr>
<tr>
<td>Total LSNs sent: 0</td>
</tr>
<tr>
<td>Total page flushes posted: 0</td>
</tr>
<tr>
<td>Total page flushes sent: 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SDS server control block: 0xc09bbd8</th>
</tr>
</thead>
<tbody>
<tr>
<td>server name: cdr_ol_nag_1_sdc2</td>
</tr>
<tr>
<td>server type: SDS</td>
</tr>
<tr>
<td>server status: Active</td>
</tr>
<tr>
<td>connection status: Connected</td>
</tr>
<tr>
<td>Last log page sent (log id, page): 7,884</td>
</tr>
<tr>
<td>Last log page flushed (log id, page): 7,884</td>
</tr>
<tr>
<td>Last LSN acked (log id, pos): 7,3621272</td>
</tr>
<tr>
<td>Sequence number of next buffer to send: 173</td>
</tr>
<tr>
<td>Sequence number of last buffer acked: 0</td>
</tr>
<tr>
<td>Time of last ack: 2007/02/20 21:04:13</td>
</tr>
<tr>
<td>Total LSNs posted: 0</td>
</tr>
<tr>
<td>Total LSNs sent: 0</td>
</tr>
<tr>
<td>Total page flushes posted: 0</td>
</tr>
<tr>
<td>Total page flushes sent: 0</td>
</tr>
</tbody>
</table>

**Figure 15-71. onstat -g sds verbose Output (run from primary server)**

**Output Description:**

- **Number of SDS servers**
  - Number of attached SDS (shared disk secondary) servers
- **Updater node alias name**
  - Name of primary server
- **SDS Server control block**
  - SD secondary server control block
- **Server types**
  - Server type
- **Server status**
  - Active or inactive
- **Connection status**
  - Status of connection between primary and secondary server
- **Last log page sent (log id, page)**
  - Log ID and page of most recent log page sent
- **Last log page flushed (lod id, page)**
  - Log ID and page of the most recent log page flushed
Last LSN acked (log id, pos)
    Most recent LSN (log position) acknowledged

Sequence number of next buffer to send
    Sequence number of next buffer to send

Sequence number of next buffer acked
    Sequence number of next buffer acknowledged

Time of last ack
    Date and time of last log acknowledgement

Total LSNs posted
    Total number of log position reports

Total LSNs sent
    Total number of log position reports sent

Total page flushes posted
    Total page flushes posted

Total page flushes sent
    Total page flushes sent

Example Output:

IBM Informix Dynamic Server Version 11.10.UC1 -- Read-Only (SDS) -- Up 00:03:17 -- 47104 Kbytes

SDS server control block: 0xb299880
Local server type: SDS
Server Status : Active
Source server name: cdr_ol_nag_1
Connection status: Connected
Last log page received (log id, page): 7,884
Next log page to read (log id, page): 7,885
Last LSN acked (log id, pos): 7,3621272
Sequence number of last buffer received: 0
Sequence number of last buffer acked: 0
Current paging file: /work1/nagaraju/dbspaces/page_cdr_ol_nag_1_sdc1_
Current paging file size: 2048
Old paging file: /work1/nagaraju/dbspaces/page_cdr_ol_nag_1_sdc1_
Old paging file size: 10240

Figure 15-72. onstat -g sds verbose Output (run from SD secondary server)

Output Description:

SDS server control block
    SD secondary server control block

Local server type
    Primary or SDS (shared disk secondary) server type

Server status
    Displays whether SD secondary server is active

Source server name
    Displays name of primary server

Connection status
    Displays whether SD secondary server is connected
Last log page received (log id, page)
   Most recent log page received

Sequence number of last buffer received
   Sequence number of last buffer received

Sequence number of last buffer acked
   Sequence number of last buffer acknowledged

Current paging file
   Name of current paging file

Current paging file size
   Size of current paging file

Old paging file
   Name of previous paging file

Old paging file size
   Size of previous paging file

**onstat -g seg: Print shared-memory segment statistics**

**Syntax:**

```bash
onstat -g seg
```

The `onstat -g seg` option prints shared-memory segment statistics. This option shows how many segments are attached and their sizes.

**Example Output:**

```
Segment Summary:
Id  key   addr   size   ovhd  class  blkused  blkfree
4001 1382438913 a000000 19918848 1760   R   4820    43
(shared) 1382438913 b2ff000 8392704 928    V   2049     0
3    1382438914 bb00000 9437184 952    V   2304     0
5    1382438915 c400000 8388608 920    V   1724    324
7    1382438916 cc00000 32505856 1656   V    7936     0
8    1382438917 eb00000 8388608 920    V    282    1766
9    1382438918 f300000 8388608 920    V    393    1655
10   1382438919 fb00000 8388608 920    V    393    1655
Total: -   -   103809024 -   -   199015443
```

(* segment locked in memory)

**Figure 15-73. onstat -g seg Output**

**onstat -g ses: Print session-related information**

**Syntax:**

```bash
onstat -g ses [sessionid]
```

Chapter 15. The onstat Utility  15-93
The `onstat -g ses` option prints session-related information. By default, only the DBSA can view `onstat -g ses` syssqltrace information. However, when `UNSECURE_ONSTAT = 1` all users can view this information. You can specify one of the following invocations.

<table>
<thead>
<tr>
<th>Invocation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>onstat -g ses</code></td>
<td>Displays a one-line summary for each session</td>
</tr>
<tr>
<td><code>onstat -g ses sessionid</code></td>
<td>Displays information for a specific session</td>
</tr>
</tbody>
</table>

```
IBM Informix Dynamic Server Version 11.10.UC1  -- On-Line -- Up 7 days 18:43:13 --
38912 Kbytes

<table>
<thead>
<tr>
<th>session</th>
<th>id</th>
<th>user</th>
<th>tty</th>
<th>pid</th>
<th>hostname</th>
<th>#RSAM</th>
<th>threads</th>
<th>total</th>
<th>used</th>
<th>dynamic</th>
<th>explain</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>24</td>
<td>informix</td>
<td>-</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
<td>12288</td>
<td>7936</td>
<td>off</td>
<td>off</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td>informix</td>
<td>-</td>
<td>17602</td>
<td>carson</td>
<td>1</td>
<td>57344</td>
<td>48968</td>
<td>off</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>informix</td>
<td>-</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>12288</td>
<td>9168</td>
<td>off</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>informix</td>
<td>-</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>12288</td>
<td>7936</td>
<td>off</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

*Figure 15-74. onstat -g ses Output*
You can interpret the output from this option as follows:

**Session section**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session id</td>
<td>The session ID</td>
</tr>
<tr>
<td>user</td>
<td>The username who started the session</td>
</tr>
<tr>
<td>tty</td>
<td>The tty associated with the front-end for this session</td>
</tr>
<tr>
<td>pid</td>
<td>The process ID associated with the front-end for this session</td>
</tr>
<tr>
<td>hostname</td>
<td>The hostname from which this session has connected</td>
</tr>
<tr>
<td>#RSAM threads</td>
<td>The number of RSAM thread allocated for this session</td>
</tr>
<tr>
<td>total memory</td>
<td>The amount of memory allocated for this session</td>
</tr>
</tbody>
</table>

**Memory pools**

<table>
<thead>
<tr>
<th>Name</th>
<th>Class</th>
<th>Address</th>
<th>Total Size</th>
<th>Free Size</th>
<th>Alloc Frag</th>
<th>Free Frag</th>
</tr>
</thead>
<tbody>
<tr>
<td>scb</td>
<td>V</td>
<td>afe0a020</td>
<td>81920</td>
<td>10200</td>
<td>119</td>
<td>13</td>
</tr>
</tbody>
</table>

**Memory pools info**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>overhead</td>
<td>0</td>
</tr>
<tr>
<td>opentable</td>
<td>0</td>
</tr>
<tr>
<td>log</td>
<td>0</td>
</tr>
<tr>
<td>keys</td>
<td>0</td>
</tr>
<tr>
<td>gentcb</td>
<td>0</td>
</tr>
<tr>
<td>sqscb</td>
<td>0</td>
</tr>
<tr>
<td>rdahead</td>
<td>0</td>
</tr>
<tr>
<td>osenv</td>
<td>0</td>
</tr>
<tr>
<td>fragman</td>
<td>0</td>
</tr>
<tr>
<td>xatm</td>
<td>0</td>
</tr>
</tbody>
</table>

**SQSCB info**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>scb</td>
<td>0</td>
</tr>
<tr>
<td>sqscb</td>
<td>0</td>
</tr>
<tr>
<td>optofc</td>
<td>0</td>
</tr>
<tr>
<td>pdqpriority</td>
<td>0</td>
</tr>
<tr>
<td>sqlstats</td>
<td>0</td>
</tr>
<tr>
<td>optcompind</td>
<td>0</td>
</tr>
<tr>
<td>directives</td>
<td>1</td>
</tr>
</tbody>
</table>

**Sess SQL**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sess SQL</td>
<td>0</td>
</tr>
<tr>
<td>Current</td>
<td>0</td>
</tr>
<tr>
<td>Iso Lock</td>
<td>0</td>
</tr>
<tr>
<td>SQL ISAM F.E.</td>
<td>0</td>
</tr>
</tbody>
</table>

**Last parsed SQL statement**

EXECUTE FUNCTION xad2pc_mi_unregister("xads_t2_i2")

**XADatasources participated in this session**

<table>
<thead>
<tr>
<th>XADatasource name</th>
<th>RMID</th>
<th>Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>xabasicdb@atmol10:sitaramv.xads_t3_i1</td>
<td>6</td>
<td>YES</td>
</tr>
<tr>
<td>xabasicdb@atmol10:sitaramv.xads_t2_i1</td>
<td>4</td>
<td>YES</td>
</tr>
<tr>
<td>xabasicdb@atmol10:sitaramv.xads_t1_i3</td>
<td>3</td>
<td>YES</td>
</tr>
<tr>
<td>xabasicdb@atmol10:sitaramv.xads_t1_i2</td>
<td>2</td>
<td>YES</td>
</tr>
<tr>
<td>xabasicdb@atmol10:sitaramv.xads_t1_i1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>xabasicdb@atmol10:sitaramv.xads_t2_i2</td>
<td>5</td>
<td>NO</td>
</tr>
</tbody>
</table>

Figure 15-75. onstat -g ses sessionid Output
used memory  The amount of memory actually used by this session

dynamic explain
Generate explain output of the sql statements of the session (on or off)

Threads section

tid      The thread ID
name     The name of the thread
rstcb    RSAM control block
flags    Describes the status of the thread using the following codes:

Position 1
B       Waiting on a buffer
C       Waiting on a checkpoint
G       Waiting on a logical-log buffer write
L       Waiting on a lock
S       Waiting on a mutex
T       Waiting on a transaction
X       Waiting on a transaction cleanup
Y       Waiting on a condition

Position 2
*      An asterisk in this position means that the thread
       encountered an I/O failure in the middle of a transaction

Position 3
A       Archive thread
B       Begin work
P       Begin Prepare or Prepared work
X       XA prepared
C       Committing or committed
R       Aborting or aborted
H       Heuristically aborted or heuristically rolling back

Position 4
P       Primary thread

Position 5
R       Reading
X       Critical section

Position 6
R       Recovery thread

Position 7
M       Monitor thread
D Daemon thread  
C Cleaner  
F Flusher  
B B-tree scanner

curstk Current stack size  
status Current thread status

**Memory pools header section.** The information is repeated for each session pool.

- **name** Name of pool  
- **class** Class of the memory where the pool is allocated from. R is for Resident, V is for Virtual, and M is for Message  
- **addr** Address of the pool structure  
- **totalsize** Total size of the memory acquired by the pool in bytes  
- **freesize** Number of bytes free in the pool  
- **#allocfrag** Number of allocated memory fragments in the pool  
- **#freefrag** Number of free fragments in the pool

**The memory pool section**

- **name** Name of a component which has allocated memory from the pool  
- **free** Number of bytes freed  
- **used** Number of bytes allocated

**The sqscb information section**

- **scb** The session control block. This is the address of the main session structure in shared memory.  
- **sqscb** SQL level control block of the session  
- **optofc** The current value of the OPTOF C environment variable or onconfig setting.  
- **pdqpriority** The current value of the PDQ PRIORITY environment variable or onconfig setting.  
- **sqlstats** The current value of the SQLSTAT S environment variable or onconfig setting.  
- **optcompind** The current value of the OPTCOMPIND environment variable or onconfig setting.  
- **directives** The current value of the DIRECTIVES environment variable or onconfig setting.

The Last parsed SQL statement section has the same information as the onstat -g sql option. See "onstat -g sql: Print SQL-related session information" on page 15-102.

**Xadatasources participated in this session section** shows information about the XA data sources that are available during the session, their resource manager identifiers, and whether they are currently active.
**X datasource name**
The XA data source that participated in the session

**RMID**
The identifier of the resource manager for the corresponding XA data source

**Active**
Whether the XA data source is still active

### onstat -g sle: Print all sleeping threads

**Syntax:**
```
  onstat -g sle
```

The `onstat -g sle` option prints all sleeping threads.

**Example Output:**

<table>
<thead>
<tr>
<th>tid</th>
<th>v_proc</th>
<th>rstcb</th>
<th>name</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>49</td>
<td>1</td>
<td>b3b13a8</td>
<td>onmode_mon</td>
<td>0.02</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>Cosvr Avail Mgr</td>
<td>0.05</td>
</tr>
<tr>
<td>42</td>
<td>1</td>
<td>b3ad028</td>
<td>main_loop()</td>
<td>0.08</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>b3ad6e8</td>
<td>xtm_svcc</td>
<td>0.64</td>
</tr>
<tr>
<td>14</td>
<td>5</td>
<td>0</td>
<td>mgmt_thd_5</td>
<td>0.65</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>0</td>
<td>mgmt_thd_4</td>
<td>0.65</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>mgmt_thd_1</td>
<td>0.65</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>0</td>
<td>dfm_svc</td>
<td>0.98</td>
</tr>
<tr>
<td>33</td>
<td>13</td>
<td>0</td>
<td>mgmt_thd_13</td>
<td>1.54</td>
</tr>
<tr>
<td>27</td>
<td>10</td>
<td>0</td>
<td>mgmt_thd_10</td>
<td>1.54</td>
</tr>
<tr>
<td>21</td>
<td>7</td>
<td>0</td>
<td>mgmt_thd_7</td>
<td>1.54</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>0</td>
<td>mgmt_thd_3</td>
<td>1.76</td>
</tr>
<tr>
<td>29</td>
<td>11</td>
<td>0</td>
<td>mgmt_thd_11</td>
<td>1.76</td>
</tr>
<tr>
<td>23</td>
<td>8</td>
<td>0</td>
<td>mgmt_thd_8</td>
<td>2.08</td>
</tr>
<tr>
<td>31</td>
<td>12</td>
<td>0</td>
<td>mgmt_thd_12</td>
<td>2.08</td>
</tr>
<tr>
<td>35</td>
<td>14</td>
<td>0</td>
<td>mgmt_thd_14</td>
<td>2.98</td>
</tr>
<tr>
<td>19</td>
<td>6</td>
<td>0</td>
<td>mgmt_thd_6</td>
<td>3.00</td>
</tr>
<tr>
<td>25</td>
<td>9</td>
<td>0</td>
<td>mgmt_thd_9</td>
<td>3.00</td>
</tr>
<tr>
<td>37</td>
<td>3</td>
<td>0</td>
<td>sch_rgm</td>
<td>3.48</td>
</tr>
<tr>
<td>44</td>
<td>5</td>
<td>b3af8a8</td>
<td>btscanner</td>
<td>7.31</td>
</tr>
<tr>
<td>46</td>
<td>3</td>
<td>b3b0628</td>
<td>bum_sched</td>
<td>41.26</td>
</tr>
</tbody>
</table>

**Figure 15-76. onstat -g sle Output**

### onstat -g smb: Print sbspaces information

**Syntax:**
```
  onstat -g smb c
  onstat -g smb fdd
```

The `onstat -g smb` option prints detailed information about sbspaces:

<table>
<thead>
<tr>
<th>Invocation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>onstat -g smb c</td>
<td>lists all the chunks in the sbspace</td>
</tr>
<tr>
<td>onstat -g smb fdd</td>
<td>lists the smart-large-object file descriptors</td>
</tr>
</tbody>
</table>
onstat -g smb lod lists the smart-large-object headers in the header table.

onstat -g smb s lists the sbspace attributes (owner, name, page size, -Df flag settings). Fields with a value of 0 or -1 were not initialized during sbspace creation.

The onstat -g smb c command displays for each sbspace chunk the following:
- chunk number and sbspace name
- chunk size and pathname
- total user data pages and free user data pages
- location and number of pages in each user-data and metadata areas

Use the onstat -g smb c option to monitor the amount of free space in each sbspace chunk, and the size in pages of the user-data, metadata, and reserved areas. In the following example, chunk 2 of sbspace1 has 2253 used pages (usr pgs) and 2245 free pages (free pg). For the first user-data area Ud1, the starting page offset is 53 and the number of pages is 1126. For the metadata area Md, the starting page offset is 1179 and the number of pages is 194. For the reserved data Ud2, the starting page offset is 1373 and the number of pages is 1127.

Chunk Summary:

<table>
<thead>
<tr>
<th>sbspace</th>
<th>chunk</th>
<th>address</th>
<th>flags</th>
<th>offset</th>
<th>size</th>
<th>orig</th>
<th>fr</th>
<th>usr pgs</th>
<th>free pg</th>
<th>path:</th>
<th>start pg</th>
<th>npages</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>303cf2a8</td>
<td>F-------</td>
<td>0</td>
<td>2500</td>
<td>2253</td>
<td>2253</td>
<td>2245</td>
<td></td>
<td>/usr11/myname/sbspace1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ud1: 53 1126
Md: 1179 194
Ud2: 1373 1127

The onstat -g smb s command displays the storage attributes for all sbspaces in the system:
- sbspace name, flags, owner
- logging status
- average smart-large-object size
- first extent size, next extent size, and minimum extent size
- maximum I/O access time
- lock mode

For more information on the onstat -g smb option, see the Performance Guide.

**onstat -g smx: Print multiplexer group information**

**Syntax:**

```
>>> onstat -g smx ses
```

The onstat -g smx command prints server multiplexer group information for servers using SMX.

<table>
<thead>
<tr>
<th>Invocation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>onstat -g smx</td>
<td>Displays SMX connection statistics</td>
</tr>
<tr>
<td>Invocation</td>
<td>Explanation</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>onstat -g smx ses</td>
<td>Displays SMX session statistics</td>
</tr>
</tbody>
</table>

Example Output:

```plaintext
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line (Prim) -- Up 00:08:06 -- 47104 Kbytes

SMX connection statistics:
SMX control block: 0x10b01c028
Peer server name: serv1_c1
SMX connection address: 0x10c2570d0
Encryption status: Enabled
Total bytes sent: 2758764
Total bytes received: 1608
Total buffers sent: 756
Total buffers received: 36
Total write calls: 95
Total read calls: 36
Total retries for write call: 1
```

Figure 15-77. onstat -g smx Output

Output Description:

- **SMX control block**
  - SMX control block

- **Peer server name**
  - Displays the name of the peer server

- **Encryption status**
  - Displays whether encryption is enabled or disabled

- **Total bytes sent**
  - Displays the total number of bytes sent

- **Total bytes received**
  - Displays the total number of bytes received

- **Total buffers sent**
  - Displays the total number of buffers sent

- **Total buffers received**
  - Displays the total number of buffers received

- **Total write calls**
  - Displays the total number of write calls

- **Total read calls**
  - Displays the total number of read calls

- **Total retries for write call**
  - Displays the total number of retries for write call

Example Output:
SMX control block

Peer name
Displays the name of the peer server

SMX session address
SMX session address

Client type
Displays type of secondary server

reads Displays the total number of session reads

writes Displays the total number of session writes

**onstat -g spi: Print spin locks with long spins**

**Syntax:**
```bash
onstat -g spi
```

The `onstat -g spi` option prints spin locks with long spins.

Many resources in the server are accessed by two or more threads. In some of these accesses (such as updating a shared value), the server must guarantee that only one thread is accessing the resource at a time. A spin lock is the mechanism used to provide this mutually exclusive access for some resources. With this type of lock, a thread that did not succeed in acquiring the lock on the first try (because another thread was holding it) repeatedly attempts to acquire the lock until it succeeds.

The overhead cost of a spin lock is small, and spin locks are normally used for resources that require mutual exclusion for short periods of time. However, if a spin lock becomes highly contended, the loop-and-retry mechanism can become expensive.

The `onstat -g spi` option is helpful for identifying performance bottlenecks that are caused by highly contended spin locks. This option lists spin locks with waits, those spin locks for which a thread was not successful in acquiring the lock on its first attempt and thus had to loop and re-attempt.

**Example Output:**
Spin locks with waits:

<table>
<thead>
<tr>
<th>Num Waits</th>
<th>Num Loops</th>
<th>Avg Loop/Wait</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1599</td>
<td>1057677</td>
<td>661.46</td>
<td>mtcb sleeping_lock</td>
</tr>
<tr>
<td>118</td>
<td>2597</td>
<td>22.01</td>
<td>class cl_lock, class = 0</td>
</tr>
<tr>
<td>1812</td>
<td>339183</td>
<td>187.19</td>
<td>vproc vp_lock, id = 1</td>
</tr>
<tr>
<td>290</td>
<td>37055</td>
<td>127.78</td>
<td>vproc vp_lock, id = 3</td>
</tr>
<tr>
<td>290</td>
<td>38117</td>
<td>315.02</td>
<td>vproc vp_lock, id = 4</td>
</tr>
<tr>
<td>1</td>
<td>25</td>
<td>25.00</td>
<td>mutex lock, name = vpc aio</td>
</tr>
<tr>
<td>10</td>
<td>16750</td>
<td>1675.00</td>
<td>shmcb sh_lock</td>
</tr>
</tbody>
</table>

**Output description:**

*Num Waits (decimal)*

Total number of times a thread waited for this spin lock.

*Num Loops (decimal)*

Total number of attempts before a thread successfully acquired the spin lock.

*Avg Loop/Wait (floating point)*

Average number of attempts needed to acquire the spin lock. Computed as Num Loops / Num Waits.

*Name (string)*

Name of the spin lock.

**onstat -g sql: Print SQL-related session information**

**Syntax:**

```
  onstat -g sql [sessionid]
```

The `onstat -g sql` option prints SQL-related information about a session. By default, only the DBSA can view `onstat -g sql` syssqltrace information. However, when UNSECURE_ONSTAT = 1 all users can view this information. You can specify one of the following invocations.

**Invocation**

**Explanation**

- `onstat -g sql`
  
  Displays a one line summary for each session

- `onstat -g sql sessionid`
  
  Displays SQL information for a specific session

**Note:** Encrypted passwords and password hint parameters in encryption functions are not shown. Figure 15-80 on page 15-103 displays an encrypted password in the Last parsed SQL statement field.
onstat -g sql 22
IBM Informix Dynamic Server Version 11.10.UC1  -- On-Line -- Up 00:07:38 -- 19456 Kbytes
Sess SQL Current Iso Lock SQL ISAM F.E. Current
Id Stmt type Database Lvl Mode ERR ERR Vers Explain Role
22 - test CR Not Wait 0 0 9.03 Off hr

Last parsed SQL statement:
select id, name, decrypt_char(ssn, 'XXXXXXXXXX') from emp

Figure 15-80. onstat -g sql Output

Output description:
Sess id     The session identifier
SQL Stmt type The type of SQL statement
Current Database Name of the current database of the session
ISO Lvl     Isolation level
     DR    Dirty Read
     CR    Committed Read
     CS    Cursor Stability
     DRU   Dirty Read, Retain Update Locks
     CRU   Committed Read, Retain Update Locks
     CSU   Cursor Stability, Retain Update Locks
     LC    Committed Read, Last Committed
     RR    Repeatable Read
     NL    Database Without Transactions
Lock mode   Lock mode of the current session
SQL Error   SQL error number encountered by the current statement
ISAM Error  ISAM error number encountered by the current statement
F.E. Version The version of the SQLI protocol used by the client program
Explain     SET EXPLAIN setting
Current Role Role of the current user

onstat -g src: Patterns in shared memory

Syntax:

onstat -g src <pattern> <mask>

The onstat -g src option searches for patterns in shared memory.

Example Output:

The following example shows output for the onstat -g src <pattern> <mask>
command where pattern = 0x123 and mask = 0xffff.
Search Summary:
addr contents
000000000ad17a50: 01090000 00000000 00000000 00000123 ........ .......
000000000ad7dec0: 00000001 014e3a0c 00000000 0ade0123 ....N.: .......

Figure 15-81. onstat -g src Output

Output description:
addr (hexidecimal)
Address in shared memory where search pattern is found
contents (hexidecimal)
Contents of memory at given address

onstat -g ssc: Print SQL statement occurrences

Syntax:

```
$ onstat -g ssc
```

The onstat -g ssc command monitors the number of times that the database server reads the SQL statement in the cache. By default, only the DBSA can view onstat -g ssc syssqltrace information. However, when UNSECURE_ONSTAT = 1 all users can view this information.

Example Output:
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:08:26 -- 29696 Kbytes

Statement Cache Summary:

#lrus  currsize  maxsize  Poolsize  #hits  nolimit
4  117640  524288  139264  0  1

Statement Cache Entries:

<table>
<thead>
<tr>
<th>lru</th>
<th>hash</th>
<th>ref_cnt</th>
<th>hits</th>
<th>flag</th>
<th>heap_ptr</th>
<th>database</th>
<th>user</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>262</td>
<td>0</td>
<td>7</td>
<td>-F</td>
<td>aad038</td>
<td>sssci007</td>
<td>admin</td>
</tr>
<tr>
<td>1</td>
<td>134</td>
<td>0</td>
<td>15</td>
<td>-F</td>
<td>aae0c38</td>
<td>sssci007</td>
<td>admin</td>
</tr>
<tr>
<td>2</td>
<td>93</td>
<td>0</td>
<td>7</td>
<td>-F</td>
<td>aae9838</td>
<td>sssci007</td>
<td>admin</td>
</tr>
<tr>
<td>2</td>
<td>276</td>
<td>0</td>
<td>7</td>
<td>-F</td>
<td>aae0c38</td>
<td>sssci007</td>
<td>admin</td>
</tr>
<tr>
<td>2</td>
<td>240</td>
<td>1</td>
<td>7</td>
<td>-F</td>
<td>b322c38</td>
<td>sssci007</td>
<td>admin</td>
</tr>
<tr>
<td>3</td>
<td>31</td>
<td>0</td>
<td>7</td>
<td>-F</td>
<td>aae0c38</td>
<td>sssci007</td>
<td>admin</td>
</tr>
<tr>
<td>3</td>
<td>45</td>
<td>0</td>
<td>1</td>
<td>-F</td>
<td>b31e438</td>
<td>sssci007</td>
<td>admin</td>
</tr>
<tr>
<td>3</td>
<td>116</td>
<td>0</td>
<td>0</td>
<td>-F</td>
<td>b362038</td>
<td>sssci007</td>
<td>admin</td>
</tr>
</tbody>
</table>

SELECT t1_char, t1_short, t1_key, t1_float, t1_smallfloat, t1_decimal, t1_serial FROM ssc1 WHERE t1_key = ?

SELECT COUNT(*) FROM ssc1

Total number of entries: 10.

Figure 15-82. onstat -g ssc Output

Output Description:

Statement Cache Summary section

#lrus Number of least recently used queues (LRUS)
currsize Current cache size
maxsize Limit on total cache memory
Poolsize Total pool size
#hits The number of hits before insertion. This number equals the value of the STMT_CACHE_HITS configuration parameter
nolimit The value of the STMT_CACHE_NOLIMIT configuration parameter

The Statement Cache Entries section shows the entries that are fully inserted into the cache.

lru The index of lru queue to which the cache entry belongs
hash Hash values of cached entry
ref_count Number of threads referencing the statement
hits Number of times a statement matches a statement in the cache. The match can be for a key-only or fully cached entry.
Cache entry flag
- F indicates the statement is fully cached
- D indicates the statement is dropped

heap_ptr Address of memory heap for cache entry

**onstat -g stk tid:** Print thread stack

**Syntax:**

```
>>onstat- g-stk-tid
```

The **onstat -g stk tid** option prints the stack of the thread specified by thread ID.

**Example Output:**

```
Stack for thread: 2 adminthd
base: 0x000000010aad5028
len: 33280
pc: 0x00000001002821e8
tos: 0x000000010aad621
state: running
vp: 2
0x1002821e8 oninit :: yield_processor + 0x260 sp=0x10aadce20(0x10ac834d0, 0x0, 0x1,
0x100000000, 0xc8a000, 0x100ba000)
0x100274e38 oninit :: wake_periodic + 0xdc sp=0x10aadced0 delta_sp=176(0x10a1b0, 0xc7a024bc,
0x0, 0x41c4, 0x10aacf98, 0x90)
0x100274fcc oninit :: admin_thread + 0x108 sp=0x10aadcf80 delta_sp=176(0x0, 0x2328,
0xd26cc0, 0x5, 0x2328a00, 0x156c)
0x1002484ec oninit :: startup + 0xd8 sp=0x10aad500 delta_sp=208(0xa, 0x10aad47d0,
0x10aad47d0, 0x100db1988, 0xda1dc80, 0x1)
```

Figure 15-83. onstat -g stk Output

**onstat -g stm: Print SQL statement memory usage**

**Syntax:**

```
>>onstat- g-stm
```

The **onstat -g stm** command displays the memory that each prepared SQL
statement uses. By default, only the DBSA can view **onstat -g stm** syssqltrace
information. However, when UNSECURE_ONSTAT = 1 all users can view this
information. To display the memory for only one session, specify the session ID in
the **onstat -g stm** option.

**Example Output:**
Output Description:

- **sdblock**: Address of the statement descriptor block
- **heapsz**: Size of the statement memory heap
- **statement**: Query text

### onstat -g stq: Print queue information

#### Syntax:

```
$ onstat -g stq [session id] [options]
```

The `onstat -g stq` command prints queue information. To view queue information for a particular session, specify the `session` option. To view queue information for all sessions, do not specify the `session` option.

#### Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:18:26 -- 6 7584 Kbytes

Stream Queue: (session 25 cnt 4) 0:db12400 1:db10400 2:dcf0400 3:dcf6400
Full Queue: (cnt 2 waiters 0) 0:0 1:db12400
Empty Queue: (cnt 0 waiters 0)
```

Figure 15-85. onstat -g stq Output

Output Description:

- **session**: Session id
- **cnt**: Number of stream queue buffers
- **waiters**: Number of threads waiting for the stream queue buffer

### onstat -g sts: Print stack usage per thread

#### Syntax:

```
$ onstat -g sts [options]
```

The `onstat -g sts` command prints maximum and current stack use per thread.

#### Example Output:

```
```

Chapter 15. The onstat Utility
Stack usage:

<table>
<thead>
<tr>
<th>TID</th>
<th>Total</th>
<th>Max</th>
<th>Current</th>
<th>Thread</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>bytes</td>
<td>%</td>
<td>bytes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>32768</td>
<td>9</td>
<td>3079</td>
<td>adminthd</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>32768</td>
<td>8</td>
<td>2871</td>
<td>childthd</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>14871</td>
<td>45</td>
<td>2871</td>
<td>Cosvr Avail Mgr</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2870</td>
<td>8</td>
<td>2871</td>
<td>dfm_svc</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3190</td>
<td>9</td>
<td>3191</td>
<td>xmf_svc</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>3580</td>
<td>9</td>
<td>3335</td>
<td>xtm_svcp</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>3238</td>
<td>9</td>
<td>3239</td>
<td>cfgmgr_svc</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>6484</td>
<td>19</td>
<td>2871</td>
<td>lio vp 0</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>6484</td>
<td>19</td>
<td>2871</td>
<td>aio vp 0</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>6484</td>
<td>19</td>
<td>2871</td>
<td>aio vp 0</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>10391</td>
<td>7</td>
<td>2871</td>
<td>msc vp 0</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>4964</td>
<td>15</td>
<td>2871</td>
<td>fifo vp 0</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>4964</td>
<td>15</td>
<td>2871</td>
<td>fifo vp 1</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>6028</td>
<td>18</td>
<td>2871</td>
<td>aio vp 1</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>5444</td>
<td>16</td>
<td>2951</td>
<td>dfmxpl_svc</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>2886</td>
<td>8</td>
<td>2887</td>
<td>sch_svc</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>7812</td>
<td>23</td>
<td>5015</td>
<td>rqm_svc</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>7140</td>
<td>21</td>
<td>3079</td>
<td>sm_poll</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>11828</td>
<td>36</td>
<td>6439</td>
<td>sm_listen</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>2870</td>
<td>8</td>
<td>2871</td>
<td>sm_discon</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>14487</td>
<td>44</td>
<td>4055</td>
<td>main_loop()</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>4272</td>
<td>13</td>
<td>2903</td>
<td>flush_sub0</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>2902</td>
<td>8</td>
<td>2903</td>
<td>flush_sub1</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>2870</td>
<td>8</td>
<td>2871</td>
<td>btscanner 0</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>3238</td>
<td>9</td>
<td>3239</td>
<td>aslogflush</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>3055</td>
<td>9</td>
<td>2887</td>
<td>bum_local</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>3238</td>
<td>9</td>
<td>3239</td>
<td>bum_rcv</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>4903</td>
<td>14</td>
<td>4903</td>
<td>onmode_mon</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>4964</td>
<td>15</td>
<td>2871</td>
<td>lio vp 1</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>5136</td>
<td>15</td>
<td>2871</td>
<td>pio vp 1</td>
<td></td>
</tr>
</tbody>
</table>

Figure 15-86. onstat -g sts Output

onstat -g sync: Print ER synchronization status

Syntax:

```
  onstat -g sync
```

The onstat -g sync command displays the synchronization status when Enterprise Replication is used. The onstat -g sync option is used primarily as a debugging tool and by IBM Support.

Example Output:
Figure 15-87. onstat -g sync Output

**Output Description:**

*Prim Repl*
Replicate number of the replicate being synchronized

*Sync Source*
Source server of the sync

*St*
Sync replicate state

*Shadow Repl*
The shadow replicate used to perform the sync

*Flag*
Internal flags:
- 0x02 = external sync
- 0x04 = shutdown request has been issued
- 0x08 = abort has occurred
- 0x010 = a replicate stop has been requested
- 0x020 = shadow or primary replicate has been deleted

*Stat*
Resync job state

*Block num*
Last block applied on targets (on source always 0)

*EndBlock Num*
Last block in resync process. Marks the end of the sync scan on the target. A value of -2 indicates that the scan is still in progress, and the highest block number is not yet known.

Additional fields for forwarded rows:

*ServID*
Server where forwarded row originated

*fwdLog ID*
Originator's log ID of the forwarded row

*fwdLog POS*
Originator's log position of the forwarded row

*endLog ID*
Operation switches back to normal at this point

*endLog POS*
Operation switches back to normal at this log position

*complete flag*
Set to 1 after normal processing resumes for the originating source.
onstat -g tpf  

**onstat -g tpf tid: Print thread profiles**

**Syntax:**

```
> onstat -g tpf tid
```

Prints thread profile for tid; 0 prints profiles for all threads.

**Example Output:**

```
onstat -g tpf 945
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:21:39 -- 29696 Kbytes
Thread profiles
  tid lkreqs lkw dl to lgrs isrd iswr isrw isdl isct isrb lx bfr bfw lsus lsmx seq
945  1969  0  0  0  6181  1782  2069  13  0  0  0  0  16183  7348  743580  0  6
```

*Figure 15-88. onstat -g tpf Output*

**Output Description:**

- **tid**: Thread ID
- **lkreqs**: Lock requests
- **lkw**: Lock waits
- **dl**: Deadlocks
- **to**: Remote deadlock timeout
- **lgrs**: Log records
- **isrd**: Number of reads
- **iswr**: Number of writes
- **isrw**: Number of rewrites
- **isdl**: Number of deletes
- **isct**: Number of commits
- **isrb**: Number of rollbacks
- **lx**: Long transactions
- **bfr**: Buffer reads
- **bfw**: Buffer writes
- **lsus**: Log space currently used
- **lsmx**: Max log space used
- **seq**: Sequence scans

**onstat -g ufr: Print memory pool fragments**

**Syntax:**

```
> onstat -g ufr
```
The `onstat -g ufr` option displays a list of the fragments that are currently in use in the specified memory pool. It requires an additional argument specifying either a pool name or session ID whose pool is to be displayed. Use `onstat -g mem` to identify the pool name and `onstat -g ses` to identify the session ID.

Memory pools are broken into fragments for various uses. With the `onstat -g ufr` command it is possible to see a list of these fragments showing their respective sizes in bytes and the type of information they contain. The information provided is generally used by IBM Support to assist in the analysis of a reported problem.

**Example Output:**

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory usage for pool name btscanner_0:</td>
<td></td>
</tr>
<tr>
<td>size memid</td>
<td></td>
</tr>
<tr>
<td>3256 overhead</td>
<td></td>
</tr>
<tr>
<td>144 scb</td>
<td></td>
</tr>
<tr>
<td>552 opentable</td>
<td></td>
</tr>
<tr>
<td>552 hashfiletab</td>
<td></td>
</tr>
<tr>
<td>2904 ostcb</td>
<td></td>
</tr>
<tr>
<td>1584 gentcb</td>
<td></td>
</tr>
<tr>
<td>12096 log</td>
<td></td>
</tr>
<tr>
<td>1912 sqtcb</td>
<td></td>
</tr>
</tbody>
</table>

*Figure 15-89. onstat -g ufr Output for pool name btscanner_0*

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory usage for pool name 6:</td>
<td></td>
</tr>
<tr>
<td>size memid</td>
<td></td>
</tr>
<tr>
<td>3256 overhead</td>
<td></td>
</tr>
<tr>
<td>144 scb</td>
<td></td>
</tr>
<tr>
<td>2968 ostcb</td>
<td></td>
</tr>
<tr>
<td>18896 sqscb</td>
<td></td>
</tr>
<tr>
<td>3312 opentable</td>
<td></td>
</tr>
<tr>
<td>72 sql</td>
<td></td>
</tr>
<tr>
<td>808 filetable</td>
<td></td>
</tr>
<tr>
<td>352 fragman</td>
<td></td>
</tr>
<tr>
<td>552 hashfiletab</td>
<td></td>
</tr>
<tr>
<td>1584 gentcb</td>
<td></td>
</tr>
<tr>
<td>12096 log</td>
<td></td>
</tr>
<tr>
<td>2960 sqtcb</td>
<td></td>
</tr>
<tr>
<td>2928 osenv</td>
<td></td>
</tr>
<tr>
<td>720 keys</td>
<td></td>
</tr>
<tr>
<td>224 rdahead</td>
<td></td>
</tr>
<tr>
<td>16248 temprec</td>
<td></td>
</tr>
</tbody>
</table>

*Figure 15-90. onstat -g ufr Output for session ID 6*

**Output Description:**

- **size (decimal)**
  - Size of the fragment in bytes

- **memid (string)**
  - Name assigned to this fragment
onstat -g vpcache: Print CPU VP memory block cache statistics

Syntax:

```
>>onstat -g vpcache
```

The `onstat -g vpcache` option returns information about CPU VP memory block cache statistics.

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 00:00:38 -- 18432 Kbytes
CPU VP memory block cache statistics - 4096 byte blocks

Number of 4096 byte memory blocks requested for each CPU VP:250

<table>
<thead>
<tr>
<th>vpid</th>
<th>pid</th>
<th>Blocks held</th>
<th>Hit percentage</th>
<th>Free cache</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7889</td>
<td>193</td>
<td>77.4 %</td>
<td>21.9 %</td>
</tr>
</tbody>
</table>

Current VP total allocations from cache: 0

<table>
<thead>
<tr>
<th>size</th>
<th>cur blks</th>
<th>alloc</th>
<th>miss</th>
<th>free</th>
<th>drain</th>
<th>release</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30</td>
<td>13</td>
<td>4</td>
<td>43</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>42</td>
<td>7</td>
<td>0</td>
<td>21</td>
<td>0</td>
<td>297</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>63</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>34</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>26</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>27</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>32</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Figure 15-91. onstat -g vpcache Output

Output Description:

15-112 IBM Informix Dynamic Server Administrator’s Reference
You can interpret output from `onstat -g vpcache` as follows:

- **size**: Is the size of the memory blocks in 4096 byte blocks
- **cur blks**: Is the current number of 4096 blocks, a multiple of size
- **alloc**: Is the number of times a requestor received a block of this size
- **miss**: Is the number of times a block was requested but none were available
- **free**: Is the number of times a memory block was placed into the cache
- **drain**: Is the number of times an aged block was forced out to make room for another block
- **release**: Is the number of times the size was full and no insertion occurred

### onstat -g wai: Print wait queue thread list

**Syntax:**

```plaintext
>>>onstat -g wai
```

The `onstat -g wai` option displays a list of the threads in the system that are currently in the wait queue and not currently executing. The output is sorted by thread ID.

**Example Output:**

```

Waiting threads:

<table>
<thead>
<tr>
<th>tid</th>
<th>tcb</th>
<th>rstcb</th>
<th>prty status</th>
<th>vp-class</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>46b1ea40</td>
<td>0</td>
<td>1  10 Idle</td>
<td>5lio</td>
<td>1io vp 0</td>
</tr>
<tr>
<td>3</td>
<td>46b3dc58</td>
<td>0</td>
<td>1  10 Idle</td>
<td>6pio</td>
<td>pio vp 0</td>
</tr>
<tr>
<td>4</td>
<td>46b5dc58</td>
<td>0</td>
<td>1  10 Idle</td>
<td>7aio</td>
<td>aio vp 0</td>
</tr>
<tr>
<td>5</td>
<td>46b7cc58</td>
<td>0</td>
<td>1  10 Idle</td>
<td>8msc</td>
<td>msc vp 0</td>
</tr>
<tr>
<td>6</td>
<td>46b1ed10</td>
<td>460f5028</td>
<td>1 sleeping secs: 1</td>
<td>3cpu</td>
<td>main_loop()</td>
</tr>
<tr>
<td>9</td>
<td>46d9d6e0</td>
<td>0</td>
<td>1  sleeping forever</td>
<td>1cpu</td>
<td>soctcp1st</td>
</tr>
<tr>
<td>10</td>
<td>46d70b48</td>
<td>0</td>
<td>1  sleeping forever</td>
<td>3cpu</td>
<td>sm_listent</td>
</tr>
<tr>
<td>11</td>
<td>46e5d9a0</td>
<td>0</td>
<td>1  sleeping secs: 1</td>
<td>3cpu</td>
<td>sm_discon</td>
</tr>
<tr>
<td>12</td>
<td>46e5dc70</td>
<td>460f5820</td>
<td>1 sleeping secs: 1</td>
<td>3cpu</td>
<td>flush_sub(0)</td>
</tr>
<tr>
<td>13</td>
<td>46e8a5a8</td>
<td>460f6018</td>
<td>1 sleeping secs: 1</td>
<td>3cpu</td>
<td>aslogflush</td>
</tr>
<tr>
<td>14</td>
<td>46fe8148</td>
<td>460f6810</td>
<td>1 sleeping secs: 41</td>
<td>3cpu</td>
<td>btscanner_0</td>
</tr>
<tr>
<td>15</td>
<td>46fe84a8</td>
<td>0</td>
<td>1  10 Idle</td>
<td>10aio</td>
<td>aio vp 1</td>
</tr>
<tr>
<td>16</td>
<td>46fe8778</td>
<td>460f7008</td>
<td>1 sleeping secs: 1</td>
<td>1cpu</td>
<td>onmode_mon</td>
</tr>
<tr>
<td>36</td>
<td>4751960</td>
<td>460f7ff8</td>
<td>1 sleeping seas: 253</td>
<td>3cpu</td>
<td>dbScheduler</td>
</tr>
<tr>
<td>37</td>
<td>47531c30</td>
<td>460f8f70</td>
<td>1 sleeping forever</td>
<td>4cpu</td>
<td>dbWorker1</td>
</tr>
<tr>
<td>38</td>
<td>47491028</td>
<td>460f7800</td>
<td>1 sleeping forever</td>
<td>4cpu</td>
<td>dbWorker2</td>
</tr>
</tbody>
</table>
```

*Figure 15-92. onstat -g wai Output*

**Output Description:**

- **tid (decimal)**
  - Thread ID
- **tcb (hex)**
  - In-memory address of the thread control block
rstcb (hex)
In-memory address of the RSAM thread control block

prty (decimal)
Thread priority. Higher numbers represent higher priorities

status (string)
Current status of the thread

vp-class (decimal and string)
Virtual processor integer ID of the VP on which the thread last ran,
concatenated with the name of the VP class upon which the thread runs

name (string)
Name of the thread

**onstat -g wmx: Print all mutexes with waiters**

Syntax:

```
>>>onstat -g wmx
```

The `onstat -g wmx` option prints all mutexes with waiters.

Example Output:

```
Mutexes with waiters:
mid  addr  name  holder  lkcnt  waiter  waittime
```

*Figure 15-93. onstat -g wmx Output*

**onstat -g wst: Print wait statistics for threads**

Syntax:

```
>>>onstat -g wst
```

The `onstat -g wst` option prints wait statistics for threads within the system. The WSTATS configuration parameter must be set to 1 to enable wait statistics collection. For more information, see "WSTATS" on page 1-108.

Example Output:
<table>
<thead>
<tr>
<th>Name</th>
<th>TID</th>
<th>State</th>
<th>N</th>
<th>Avg(us)</th>
<th>Max(us)</th>
</tr>
</thead>
<tbody>
<tr>
<td>lio vp 0 2</td>
<td>ready</td>
<td>128</td>
<td>17</td>
<td>454819</td>
<td></td>
</tr>
<tr>
<td>lio vp 0 2</td>
<td>run</td>
<td>127</td>
<td>405</td>
<td>441</td>
<td></td>
</tr>
<tr>
<td>lio vp 0 2</td>
<td>10 Idle</td>
<td>126</td>
<td>643203</td>
<td>727160</td>
<td></td>
</tr>
<tr>
<td>pio vp 0 3</td>
<td>yield 0</td>
<td>1</td>
<td>38</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>pio vp 0 3</td>
<td>ready</td>
<td>2</td>
<td>22</td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>pio vp 0 3</td>
<td>run</td>
<td>1</td>
<td>16</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>aio vp 0 4</td>
<td>yield 0</td>
<td>1</td>
<td>37</td>
<td>37</td>
<td></td>
</tr>
<tr>
<td>aio vp 0 4</td>
<td>yield time</td>
<td>2</td>
<td>747701</td>
<td>904462</td>
<td></td>
</tr>
<tr>
<td>aio vp 0 4</td>
<td>ready</td>
<td>230</td>
<td>129</td>
<td>5284</td>
<td></td>
</tr>
<tr>
<td>aio vp 0 4</td>
<td>run</td>
<td>229</td>
<td>145</td>
<td>10045</td>
<td></td>
</tr>
<tr>
<td>aio vp 0 4</td>
<td>10 Idle</td>
<td>226</td>
<td>45823</td>
<td>941363</td>
<td></td>
</tr>
<tr>
<td>msc vp 0 5</td>
<td>yield 0</td>
<td>1</td>
<td>38</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>msc vp 0 5</td>
<td>ready</td>
<td>5</td>
<td>280</td>
<td>1273</td>
<td></td>
</tr>
<tr>
<td>msc vp 0 5</td>
<td>run</td>
<td>4</td>
<td>178</td>
<td>429</td>
<td></td>
</tr>
<tr>
<td>msc vp 0 5</td>
<td>10 Idle</td>
<td>3</td>
<td>896605</td>
<td>1.0s</td>
<td></td>
</tr>
<tr>
<td>main_loo 6</td>
<td>10 Wait</td>
<td>26</td>
<td>10274</td>
<td>12113</td>
<td></td>
</tr>
<tr>
<td>main_loo 6</td>
<td>yield time</td>
<td>6416</td>
<td>1.0s</td>
<td>1.0s</td>
<td></td>
</tr>
<tr>
<td>main_loo 6</td>
<td>yield forever</td>
<td>4</td>
<td>97377</td>
<td>105682</td>
<td></td>
</tr>
<tr>
<td>main_loo 6</td>
<td>ready</td>
<td>6450</td>
<td>23</td>
<td>31864</td>
<td></td>
</tr>
<tr>
<td>main_loo 6</td>
<td>run</td>
<td>6436</td>
<td>5</td>
<td>3500</td>
<td></td>
</tr>
<tr>
<td>soctcppo 7</td>
<td>yield forever</td>
<td>1027128</td>
<td>3</td>
<td>1.0s</td>
<td></td>
</tr>
<tr>
<td>soctcppo 7</td>
<td>other cond</td>
<td>1</td>
<td>110728</td>
<td>110728</td>
<td></td>
</tr>
<tr>
<td>soctcppo 7</td>
<td>ready</td>
<td>2</td>
<td>172080</td>
<td>1.3s</td>
<td></td>
</tr>
<tr>
<td>soctcppo 7</td>
<td>run</td>
<td>1027127</td>
<td>118377</td>
<td>1.0s</td>
<td></td>
</tr>
<tr>
<td>sm_poll 8</td>
<td>yield 0</td>
<td>1</td>
<td>61</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>sm_poll 8</td>
<td>yield time</td>
<td>1</td>
<td>887246</td>
<td>887246</td>
<td></td>
</tr>
<tr>
<td>sm_poll 8</td>
<td>ready</td>
<td>3</td>
<td>30</td>
<td>69</td>
<td></td>
</tr>
<tr>
<td>sm_poll 8</td>
<td>run</td>
<td>1</td>
<td>30</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>soctcpls 9</td>
<td>10 Wait</td>
<td>5</td>
<td>781</td>
<td>1580</td>
<td></td>
</tr>
<tr>
<td>soctcpls 9</td>
<td>ready</td>
<td>7</td>
<td>141</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>soctcpls 9</td>
<td>run</td>
<td>5</td>
<td>267</td>
<td>695</td>
<td></td>
</tr>
<tr>
<td>sm_liste 10</td>
<td>10 Wait</td>
<td>8</td>
<td>168</td>
<td>718</td>
<td></td>
</tr>
<tr>
<td>sm_liste 10</td>
<td>ready</td>
<td>9</td>
<td>93</td>
<td>629</td>
<td></td>
</tr>
<tr>
<td>sm_liste 10</td>
<td>run</td>
<td>8</td>
<td>99</td>
<td>561</td>
<td></td>
</tr>
<tr>
<td>sm_disco 11</td>
<td>yield time</td>
<td>6417</td>
<td>1.0s</td>
<td>1.0s</td>
<td></td>
</tr>
<tr>
<td>sm_disco 11</td>
<td>ready</td>
<td>6418</td>
<td>38</td>
<td>38860</td>
<td></td>
</tr>
<tr>
<td>sm_disco 11</td>
<td>run</td>
<td>6417</td>
<td>2</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>flush_su 12</td>
<td>yield time</td>
<td>6417</td>
<td>1.0s</td>
<td>1.1s</td>
<td></td>
</tr>
<tr>
<td>flush_su 12</td>
<td>ready</td>
<td>6418</td>
<td>38</td>
<td>38891</td>
<td></td>
</tr>
<tr>
<td>flush_su 12</td>
<td>run</td>
<td>6417</td>
<td>2</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>aslogflu 13</td>
<td>yield time</td>
<td>6416</td>
<td>1.0s</td>
<td>1.0s</td>
<td></td>
</tr>
<tr>
<td>aslogflu 13</td>
<td>ready</td>
<td>6418</td>
<td>33</td>
<td>38824</td>
<td></td>
</tr>
<tr>
<td>aslogflu 13</td>
<td>run</td>
<td>6417</td>
<td>2</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>btscanne 14</td>
<td>yield 0</td>
<td>1</td>
<td>7</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>btscanne 14</td>
<td>yield time</td>
<td>72</td>
<td>498264</td>
<td>623090</td>
<td></td>
</tr>
<tr>
<td>btscanne 14</td>
<td>ready</td>
<td>222</td>
<td>765502</td>
<td>1.0s</td>
<td></td>
</tr>
<tr>
<td>btscanne 14</td>
<td>run</td>
<td>73</td>
<td>123</td>
<td>653</td>
<td></td>
</tr>
<tr>
<td>onmode_m 25</td>
<td>yield time</td>
<td>6414</td>
<td>1.0s</td>
<td>1.0s</td>
<td></td>
</tr>
<tr>
<td>onmode_m 25</td>
<td>ready</td>
<td>6416</td>
<td>29</td>
<td>38816</td>
<td></td>
</tr>
<tr>
<td>onmode_m 25</td>
<td>run</td>
<td>6414</td>
<td>4</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>aio vp 1 30</td>
<td>yield 0</td>
<td>1</td>
<td>37</td>
<td>37</td>
<td></td>
</tr>
<tr>
<td>aio vp 1 30</td>
<td>ready</td>
<td>143</td>
<td>11</td>
<td>278</td>
<td></td>
</tr>
<tr>
<td>aio vp 1 30</td>
<td>run</td>
<td>142</td>
<td>11</td>
<td>142</td>
<td></td>
</tr>
<tr>
<td>aio vp 1 30</td>
<td>10 Idle</td>
<td>141</td>
<td>45023</td>
<td>779089</td>
<td></td>
</tr>
</tbody>
</table>
Output Description:

name (string)
Thread name

tid (decimal)
Thread ID

state (string)
State the thread waited in for this line of output. A single thread may have multiple lines of output if it has waited in more than one different state.

n (decimal)
Number of times the thread waited in this state

avg(us) (floating point)
Average user time the thread spent waiting in this state per wait occurrence. Time is in microseconds; an s after the value indicates user time in seconds.

max(us) (floating point)
Maximum user time the thread spent waiting in this state for a single wait occurrence. Time is in microseconds; an s after the value indicates user time in seconds.

onstat -G: Print TP/XA transaction information

Syntax:

```bash
onstat -G
```

Use the -G option to display information about global transactions generated through TP/XA. For more information on TP/XA, see the *IBM Informix TP/XA Programmer’s Manual*.

Example Output:

Figure 15-95 shows an example of `onstat -G` output:

<table>
<thead>
<tr>
<th>Global Transaction Identifiers</th>
<th>flags</th>
<th>isol</th>
<th>timeout</th>
<th>fID</th>
<th>bql</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ae35e34 -LR-G COMMIT 0</td>
<td>4478019</td>
<td>16</td>
<td>48</td>
<td>438709F23076254C80F33A62B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AF4CF763C1BCFFAD7AE0243A5CE243FA5381C93A9F52A1546044992C5A7BC035B2E77999EFBA7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25D3D408DAF37404D0AFF1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ae3623c AL--G COMMIT 0</td>
<td>4478019</td>
<td>16</td>
<td>48</td>
<td>438709F23076254C80F33A62B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AF4CF763C1BCFFAD7AE0243A5CE243FA5381C93A9F52A1546044992C5A7BC035B2E77999EFBA7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25D3D408DAF37404D0A900</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2 active, 128 total

For a tightly coupled transaction, all branches will share the same transaction address shown in the address column.

Output Description:

The flag codes for position 1 (current transaction state):

A User thread attached to the transaction
S  TP/XA suspended transaction
C  TP/XA waiting for rollback

The flag codes for position 2 (transaction mode):
T  Tightly-coupled mode (MTS)
L  Loosely-coupled mode (default mode)

The flag codes for position 3 (transaction stage):
B  Begin work
P  Distributed query prepared for commit
X  TP/XA prepared for commit
C  Committing or committed
R  Rolling back or rolled back
H  Heuristically rolling back or rolled back

The flag code for position 4:
X  XA DataSource global transaction

The flag codes for position 5 (type of transaction):
G  Global transaction
C  Distributed query coordinator
S  Distributed query subordinate
B  Both distributed query coordinator and subordinate

---

**onstat -h: Print buffer header hash chain information**

**Syntax:**

```
onstat -h
```

The `onstat -h` option provides information on the buffer header hash chains (sometimes called "hash buckets") used to access pages in each buffer pool, in the form of a numeric histogram of chain lengths, plus summary information for each buffer pool. All numeric values in the output are decimal. Shorter hash chains enable requested buffers to be located more quickly by the server, because on average it will need to check fewer buffer headers on a target chain to find the target buffer.

The page size of the buffer pool in bytes is shown as a header to the output for each buffer pool. The histogram and summary information are then presented for that buffer pool.

**Example Output:**
IBM Informix Dynamic Server Version 11.10.F -- On-Line -- Up 00:00:14 -- 1071740 Kbytes

Buffer pool page size: 2048

buffer hash chain length histogram

<table>
<thead>
<tr>
<th># of chains</th>
<th>of len</th>
</tr>
</thead>
<tbody>
<tr>
<td>3423</td>
<td>0</td>
</tr>
<tr>
<td>4546</td>
<td>1</td>
</tr>
<tr>
<td>223</td>
<td>2</td>
</tr>
<tr>
<td>8192</td>
<td>total chains</td>
</tr>
<tr>
<td>4992</td>
<td>hashed buffs</td>
</tr>
<tr>
<td>5000</td>
<td>total buffs</td>
</tr>
</tbody>
</table>

Buffer pool page size: 4096

buffer hash chain length histogram

<table>
<thead>
<tr>
<th># of chains</th>
<th>of len</th>
</tr>
</thead>
<tbody>
<tr>
<td>707</td>
<td>0</td>
</tr>
<tr>
<td>315</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>1024</td>
<td>total chains</td>
</tr>
<tr>
<td>319</td>
<td>hashed buffs</td>
</tr>
<tr>
<td>1000</td>
<td>total buffs</td>
</tr>
</tbody>
</table>

Figure 15-96. onstat -h Output

Output Description:

You can interpret output from this option as follows:

Histogram Information on Hash Chains

The histogram information has a row for each buffer hash chain
length that presently exists in the system. Each row has two
columns:

# of chains
   Number of hash chains of the given length

of len
   Length of these chains

Summary Information Per Buffer Pool

total chains
   Number of hash chains that exist for this buffer pool

hashed buffs
   Number of buffer headers currently hashed into the hash
   chains for this buffer pool

total buffs
   Total number of buffers in this buffer pool

onstat -i: Initiate interactive mode

Syntax:

>>>onstat -i

[ r—seconds ]
[ rz—seconds ]
Use the `-i` option to put `onstat` in interactive mode. In interactive mode, you can enter multiple `onstat` options per session, but only one at a time. An `onstat` prompt appears and allows you to enter an option.

In interactive mode, do not precede the option with a dash.

Two additional options, `r seconds` and `rz seconds`, are available in interactive mode. The `r seconds` option is similar to the current `onstat -r seconds` option, which repeatedly generates a display. If an administrator executes `r seconds` at the interactive-mode prompt, the prompt changes to reflect the specified interval in seconds and reappears, waiting for the next command. In the following example, the display generated by the next command repeats every three seconds:

```
onstat> r 3
onstat[3]>
```

The `rz seconds` option enables you to repeat the next command as specified and set all profile counters to 0 between each execution.

To terminate interactive mode, press CTRL-d.

To terminate a repeating sequence, press CTRL-c.

---

**onstat -j: Provide onload status information**

**Syntax:**

```
onstat -j
```

The `-j` option of the `onstat` utility provides special information about the status of an `onload` job. The `-j` option provides an interactive mode that is analogous to `onstat -i`.

When `onload` starts, it writes a series of messages to `stdout` or to a log file. The following lines show a typical `onload` log file:

```
Mon Jul 23 16:11:30 2007
SHMBASE 0x44000000
CLIENTNUM 0x49010000
Session ID 1
Load Database --> cnv001
Load Table --> cnv001a
Load File --> testrec.dat
Record Mapping --> cnv001a
Database Load Completed -- Processed 50 Records
Records Inserted--> 50
Detected Errors--> 0
Engine Rejected--> 0
Mon Jul 23 16:11:37 2007
```

The two lines that start with SHMBASE and CLIENTNUM provide the information that you need to locate shared memory for an instance of `onload`. The `oninit` process has similar values stored in the `$ONCONFIG` file. When you use `onstat` to gather information about the `oninit` process, `onstat` uses information from `$INFORMIXDIR/etc/$ONCONFIG` to locate shared memory. When you use `onstat`
to gather information about **onpload**, you must give **onstat** the name of a file that contains SHMBASE and CLIENTNUM information.

Typically the file that contains the SHMBASE and CLIENTNUM information is the log file. For example, if the **onpload** log file is `/tmp/cnv001a.log`, you can enter the following command:

```
onstat -j /tmp/cnv001a.log
```

The previous command causes **onstat** to attach to **onpload** shared memory and to enter interactive mode. You can then enter ? or any other pseudo request to see a usage message displayed. An example follows:

```
onstat> ?
Interactive Mode: One command per line, and - are optional.
   -rz repeat option every n seconds (default: 5) and
     zero profile counts
MT COMMANDS:
   all   Print all MT information
   ath   Print all threads
   wai   Print waiting threads
   act   Print active threads
   rea   Print ready threads
   sle   Print all sleeping threads
   spi   Print spin locks with long spins
   sch   Print VP scheduler statistics
   lmx   Print all locked mutexes
   wmex  Print all mutexes with waiting
   con   Print conditions with waiters
   stk <tid> Dump the stack of a specified thread
   glo   Print MT global information
   mem <pool name|session id> print pool statistics.
   seg   Print memory segment statistics.
   rbm   print block map for resident segment
   nbm   print block map for non-resident segments
   afr <pool name|session id> Print allocated poolfragments.
   ffr <pool name|session id> Print free pool fragments.
   ufr <pool name|session id> Print pool usage breakdown
   iov   Print disk IO statistics by vp
   iof   Print disk IO statistics by chunk/file
   i0q   Print disk IO statistics by queue
   iog   Print AIO global information
   iob   Print big buffer usage by IO VP class
   sts   Print max and current stack sizes
   qst   print queue statistics
   wst   print thread wait statistics
   jal   Print all Pload information
   jct   Print Pload control table
   jpa   Print Pload program arguments
   jta   Print Pload thread array
   jmq   Print Pload message queues, jms for summary only

onstat>
```

Most of the options are the same as those that you use to gather information about Dynamic Server, with the following exceptions:

```
jal   Print all Pload information
jct   Print Pload control table
jpa   Print Pload program arguments
jta   Print Pload thread array
jmq   Print Pload message queues, jms for summary only

onstat>
```
These options apply only to onpload. You can use onstat -j to check the status of a thread, locate the VP and its PID, and then attach a debugger to a particular thread. The options for onstat that do not apply to onpload are not available (for example, -g ses).

**onstat -k: Print active lock information**

**Syntax:**

```
   onstat -k
```

Use the -k option to display information about active locks.

**Example Output:**

```

Locks
  address  wtlist  owner  lklist  type  tblsnum  rowid  key# / bsiz
  a095f78  0      a4d5e68  0      HDR+S  100002  203   0

1 active, 2000 total, 2048 hash buckets, 0 lock table overflows
```

*Figure 15-97. onstat -k Output*

**Output Description:**

You can interpret output from this option as follows:

- **address**: Is the address of the lock in the lock table. If a user thread is waiting for this lock, the address of the lock appears in the wait field of the onstat -u (users) output.

- **wtlist**: Is the first entry in the list of user threads that is waiting for the lock, if there is one.

- **owner**: Is the shared-memory address of the thread that is holding the lock. This address corresponds to the address in the address field of onstat -u (users) output. When the owner value is displayed in parenthesis, it represents the shared memory address of a transaction structure. This scenario is possible only when a lock is allocated for a global transaction. This address corresponds to the address field of the output for onstat -G.

- **lklist**: Is the next lock in a linked list of locks held by the owner just listed.

- **type**: Uses the following codes to indicate the type of lock:
  - HDR: Header
  - B: Bytes
  - S: Shared
  - X: Exclusive
  - I: Intent
U Update
IX Intent-exclusive
IS Intent-shared
SIX Shared, intent-exclusive

\textit{tbsnum} Is the tblspace number of the locked resource. If the number is less than 10000, it indicates Enterprise Replication pseudo locks.

\textit{rowid} Is the row identification number.

The \textit{rowid} provides the following lock information:
- If the \textit{rowid} equals zero, the lock is a table lock.
- If the \textit{rowid} ends in two zeros, the lock is a page lock.
- If the \textit{rowid} is six digits or fewer and does not end in zero, the lock is probably a row lock.
- If the \textit{rowid} is more than six digits, the lock is probably an index key-value lock.

\textit{key/#bsiz} Is the index key number, or the number of bytes locked for a VARCHAR lock.

If this field contains ‘K-’ followed by a value, it is a key lock. The value identifies which index is being locked. For example, K-1 indicates a lock on the first index defined for the table.

The maximum number of locks available is specified as \textit{LOCKS} in the \textit{ONCONFIG} file.

---

**onstat -l: Print physical and logical log information**

**Syntax:**

```
$onstat -l
```

Use the \texttt{-l} option to display information about physical and logical logs.

**Example Output:**

---

Physical Logging
Buffer bufused bufsize numpages numwrits pages/io
P-1 0 16 716 55 13.02
phybegin physize phypos phyused %used
1:263 500 270 0 0.00

Logical Logging
Buffer bufused bufsize numrecs numpages numwrits recs/pages pages/io
L-3 0 16 42169 2872 1043 14.7 2.8
Subsystem numrecs Log Space used
OLDRSAM 42169 4436496
address number flags uniqid begin size used %used
a517f70 1 U-B----- 1 1:763 500 500 100.00
a517fd0 2 U-B----- 2 1:1263 500 500 100.00
a40daf0 3 U-B----- 3 1:1763 500 500 100.00
a40db30 4 U-B----- 4 1:2263 500 500 100.00
a40db70 5 U-B----- 5 1:2763 500 500 100.00
a40dbb0 6 U-C-L 6 1:3263 500 372 74.40
a40dbf0 7 A----- 0 1:3763 500 0 0.00
a40dc30 8 A----- 0 1:4263 500 0 0.00
8 active, 8 total

Figure 15-98. onstat -l Output

Output Description:

You can interpret output from this option as follows. The first section of the display describes the physical-log configuration:

- **buffer** Is the number of the physical-log buffer
- **bufused** Is the number of pages of the physical-log buffer that are used
- **bufsize** Is the size of each physical-log buffer in pages
- **numpages** Is the number of pages written to the physical log
- **numwrits** Is the number of writes to disk
- **pages/io** Is calculated as numpages/numwrits

This value indicates how effectively physical-log writes are being buffered.

- **phybegin** Is the physical page number of the beginning of the log
- **physize** Is the size of the physical log in pages
- **phypos** Is the current position in the log where the next log-record write is to occur
- **phyused** Is the number of pages used in the log
- **%used** Is the percent of pages used

The second section of the **onstat -l** display describes the logical-log configuration:

- **buffer** Is the number of the logical-log buffer
- **bufused** Is the number of pages used in the logical-log buffer
- **bufsize** Is the size of each logical-log buffer in pages
- **numrecs** Is the number of records written
numpages  Is the number of pages written
numwrits  Is the number of writes to the logical log
recs/pages Is calculated as numrecs/numpages
You cannot affect this value. Different types of operations generate
different types (and sizes) of records.

pages/io is calculated as numpages/numwrits
You can affect this value by changing the size of the logical-log
buffer (specified as LOGBUFF in the ONCONFIG file) or by
changing the logging mode of the database (from buffered to
unbuffered, or vice versa).

The following fields are repeated for each logical-log file:

address  Is the address of the log-file descriptor
number   Is logid number for the logical-log file
The logid numbers might be out of sequence because either the
database server or administrator can insert a log file in-line.

flags  Provides the status of each log as follows:
A  Newly added (and ready to use)
B  Backed up
C  Current logical-log file
D  Marked for deletion
   To drop the log file and free its space for reuse, you must
   perform a level-0 backup of all storage spaces
F  Free, available for use
L  The most recent checkpoint record
U  Used

uniqid  Is the unique ID number of the log
begin   Is the beginning page of the log file
size    Is the size of the log in pages
used    Is the number of pages used
%used   Is the percent of pages used
active  Is the number of active logical logs
total   Is the total number of logical logs

The database server uses temporary logical logs during a warm restore because the
permanent logs are not available then. The following fields are repeated for each
temporary logical-log file:

address  Is the address of the log-file descriptor
number   Is logid number for the logical-log file
flags    Provides the status of each log as follows:
B  Backed up
C Current logical-log file
F Free, available for use
U Used

uniqid Is the unique ID number of the log
begin Is the beginning page of the log file
size Is the size of the log in pages
used Is the number of pages used
%used Is the percent of pages used
active Is the number of active temporary logical logs

**onstat -m: Print recent system message log information**

**Syntax:**

```plaintext
>>> onstat -m
```

Use the `-m` option to display the 20 most-recent lines of the system message log. You can use the `onstat -m` option with the database server in any mode, including offline.

Output from this option lists the full pathname of the message-log file and the 20 file entries. A date-and-time header separates the entries for each day. A time stamp prefaces single entries within each day. The name of the message log is specified as MSGPATH in the `ONCONFIG` file.

**Example Output:**

```

Message Log File: /work/11.10/dbspaces/star3.log
11:26:33 Checkpoint Completed: duration was 0 seconds.
11:26:33 Checkpoint loguniq 1, logpos 0x23c408, timestamp: 0x2cc2 Interval: 9
```

*Figure 15-99. onstat -m Output*

**onstat -o: Output shared memory contents**

**Syntax:**

```plaintext
>>> onstat -o
```

Use the `onstat -o` option to copy the contents of shared memory out to a specified file for later analysis. If no output file is specified, by default `onstat.out` in the current directory is used. The file created by `onstat -o` is the same size as the shared memory segments for the Informix Dynamic Server instance. You must have enough room in the file system to handle the output.
The `onstat -o` option allows you to capture the current shared memory information to a file. By executing additional `onstat` commands against the file, you can gather information from a previously saved shared memory dump. For more information, see "`onstat <option> <infile>`: Print shared memory dump file" on page 15-9.

---

**onstat -O: Print optical subsystem information**

**Syntax:**

```
$ onstat -O
```

Use the `-O` option of the `onstat` utility to display information about the Optical Subsystem memory cache and staging-area blobspace. You can interpret output from this option as follows. The totals shown in the display accumulate from session to session. The database server resets the totals to 0 only when you execute `onstat -z`.

**Example Output:**

```
IBM Informix Dynamic Server Version 11.10.UC1 --Online-- Up 00:45:18 -- 11656 Kbytes

Optical StageBlob Cache
System Cache Totals:
Size Alloc. Avail. Number Kbytes Number Kbytes
500 500 0 1 20 3 1500

User Cache Totals:
SID User Size Number Kbytes Number Kbytes
94 doug 250 1 20 1 100
95 beth 500 0 0 2 1200
```

*Figure 15-100. onstat -O Output*

**Output Description:**

The first section of the display provides the following information on system-cache totals:

- `size`: Is the size that the OPCODEMAX configuration parameter specifies
- `alloc`: Is the number of 1-kilobyte allocations to the cache
- `avail`: Describes how much of `alloc` (in kilobytes) is not used
- `number`: Is the number of simple large objects that the database server successfully put in the cache without overflowing
- `kbytes`: Is the number of kilobytes of TEXT or BYTE data that the database server put in the cache without overflowing
- `number`: Is the number of simple large objects that the database server wrote to the staging-area blobspace
- `kbytes`: Is the number of kilobytes of TEXT or BYTE data that the database server wrote to the staging-area blobspace
Although the size output indicates the amount of memory that is specified in the configuration parameter OPCODEMAX, the database server does not allocate memory to OPCODEMAX until necessary. Therefore, the alloc output reflects only the number of 1-kilobyte allocations of the largest simple large object that has been processed. When the values in the alloc and avail output are equal to each other, the cache is empty.

The second section of the display describes the following user-cache totals information:

- **SID**: Is the session ID for the user
- **user**: Is the user ID of the client
- **size**: Is the size specified in the INFORMIXOPCACHE environment variable, if it is set
  
  If you do not set the INFORMIXOPCACHE environment variable, the database server uses the size that you specify in the configuration parameter OPCODEMAX.
- **number**: Is the number of simple large objects that the database server put into cache without overflowing
- **kbytes**: Is the number of kilobytes of TEXT or BYTE data that the database server put in the cache without overflowing
- **number**: Is the number of simple large objects that the database server wrote to the staging-area blobspace
- **kbytes**: Is the number of kilobytes of TEXT or BYTE data that the database server wrote to the staging-area blobspace

The last line of the display lists the total number of sessions that are using the cache.

---

**onstat -p: Print profile counts**

**Syntax:**

```onstat -p```

Use the -p option to display profile counts either since you started the database server or since you ran `onstat` with the -z option.

**Example Output:**

---

Chapter 15. The onstat Utility 15-127
Output Description:

The first portion of the display describes reads and writes.

Reads and writes are tabulated in three categories: from disk, from buffers, and number of pages (read or written).

The first %cached field is a measure of the number of reads from buffers compared to reads from disk. The second %cached field is a measure of the number of writes to buffers compared to writes to disk.

The database server buffers information and writes to the disk in pages. For this reason, the number of disk writes displayed as dskwrits is usually less than the number of writes that an individual user executes:

- **dskreads** is the number of actual reads from disk
- **pagreads** is the number of pages read
- **bufreads** is the number of reads from shared memory
- **%cached** is the percent of reads cached, calculated as follows:
  \[100 \times \frac{\text{bufreads} - \text{dskreads}}{\text{bufreads}}\]
  If bufreads exceeds the maximum integer (or long) value, its internal representation becomes a negative number, but the value appears as 0.0.
- **dskwrits** is the actual number of physical writes to disk
  This number includes the writes for the physical and logical logs reported in **onstat -l**.
- **pagwrits** is the number of pages written
- **bufwrits** is the number of writes to shared memory
- **%cached** is the percent of writes cached, calculated as follows:
  \[100 \times \frac{\text{bufwrits} - \text{dskwrits}}{\text{bufwrits}}\]
If dskwrits exceeds bufwrits, the value appears as 0.0. The next portion of the -p display tabulates the number of times different ISAM calls were executed. The calls occur at the lowest level of operation and do not necessarily correspond one-to-one with SQL statement execution. A single query might generate multiple ISAM calls. These statistics are gathered across the database server and cannot be used to monitor activity on a single database unless only one database is active or only one database exists:

- `isamtot`: Is the total number of calls
- `open`: Increments when a tblspace is opened
- `start`: Increments the pointer within an index
- `read`: Increments when the read function is called
- `write`: Increments with each write call
- `rewrite`: Increments when an update occurs
- `delete`: Increments when a row is deleted
- `commit`: Increments each time that an `iscommit()` call is made
- `rollbk`: Increments when a transaction is rolled back

The next portion of the -p display provides information on generic pages. The Generic Page Manager provides an API for Dynamic Server to manage nonstandard pages in the database server buffer pool. The following table describes the Generic Page Manager fields in the `onstat -p` output.

- `gp_read`: The number of generic page reads
- `gp_zWRITE`: The number of generic page writes
- `gp_reWRT`: The number of generic page updates
- `gp_del`: The number of generic page deletes
- `gp_alloc`: The number of generic page allocations
- `gp_free`: The number of generic pages freed and returned to tblspaces
- `gp_curs`: The number of cursors used against generic pages

The next portion of the -p display tracks the number of times that a resource was requested when none was available:

- `ovlock`: Is the number of times that the database server attempted to allocate locks more than 15 times
- `ovuserthread`: Is the number of times that a user attempted to exceed the maximum number of user threads
- `ovbuff`: Is the number of times that the database server could not find a free shared-memory buffer

When no buffers are free, the database server writes a dirty buffer to disk and then tries to find a free buffer.
usercpu  Is the total user CPU time that all user threads use, expressed in seconds
This entry is updated every 15 seconds.
syscpu  Is the total system CPU time that all user threads use, expressed in seconds
This entry is updated every 15 seconds.
numckpts  Is the number of checkpoints since the boot time
flushes  Is the number of times that the buffer pool has been flushed to the disk

The next portion of the -p display contains miscellaneous information, as follows:
bufwaits  Increments each time that a user thread must wait for a buffer
lokwaits  Increments each time that a user thread must wait for a lock
lockreqs  Increments each time that a lock is requested
deadlks  Increments each time that a potential deadlock is detected and prevented
dltouts  Increments each time that the distributed deadlock time-out value is exceeded while a user thread is waiting for a lock
ckpwaits  Is the number of checkpoint waits
compress  Increments each time that a data page is compressed
seqscans  Increments for each sequential scan

The last portion of the -p display contains the following information:
ixda-RA  Is the count of read-aheads that go from index leaves to data pages
idx-RA  Is the count of read-aheads that traverse index leaves
da-RA  Is the count of data-path-only scans
RA-pgsused  Indicates the number of pages used that the database server read ahead
  If this number is significantly less than the total number of pages read ahead, the read-ahead parameters might be set too high.
lchwaits  Stores the number of times that a thread was required to wait for a shared-memory latch
  A large number of latch waits typically results from a high volume of processing activity in which the database server is logging most of the transactions.

**onstat -P: Print partition information**

**Syntax:**

```
#$onstat -P
```

Use the -P option to display for all partitions the partition number and the pages in the buffer pool that belong to the partition.
Example Output:

```
Buffer pool page size: 2048

<table>
<thead>
<tr>
<th>partnum</th>
<th>total</th>
<th>btree</th>
<th>data</th>
<th>other</th>
<th>dirty</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>36</td>
<td>1</td>
<td>8</td>
<td>27</td>
<td>0</td>
</tr>
<tr>
<td>1048577</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>1048578</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>1048579</td>
<td>23</td>
<td>10</td>
<td>12</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1048580</td>
<td>68</td>
<td>31</td>
<td>36</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4194309</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Totals: 3000 786 1779 435 0

Percentages:
Data 59.30
Btree 26.20
Other 14.50

Buffer pool page size: 8192

<table>
<thead>
<tr>
<th>partnum</th>
<th>total</th>
<th>btree</th>
<th>data</th>
<th>other</th>
<th>dirty</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>999</td>
<td>0</td>
<td>0</td>
<td>999</td>
<td>0</td>
</tr>
<tr>
<td>5242881</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Totals: 1000 0 0 1000 0

Percentages:
Data 0.00
Btree 0.00
Other 100.00
```

Figure 15-102. onstat -P Output

**Output Description:**

- **Buffer pool page size**
  - Is the size of the buffer pool pages in bytes
- **partnum**
  - Is the partition number
- **total**
  - Is the total number of partitions
- **btree**
  - Is the number of B-tree pages in the partition
- **data**
  - Is the number of data pages in the partition
- **other**
  - Is the number of other pages in the partition
- **resident**
  - Is the number of resident pages in the partition
- **dirty**
  - Is the number of dirty pages in the partition

**onstat -r: Repeatedly print selected statistics**

**Syntax:**

```
|--onstat-- -r  
  |--seconds--other_options--  
    | other_options-- 
```

The **onstat -r** option causes the other options specified in the command to be repeated at specified intervals. Use **onstat -r [seconds] [other_options]** to specify the seconds to repeat the other option. Use **onstat -r [other_options]** to have the
option repeat every five seconds, which allows the other options to be concatenated with the -r flag, as in this example: onstat -rFh. The onstat -r option can be used in both command mode and interactive mode, and can be useful for repeating command output to monitor system resource utilization.

Example Output 1: execute `onstat` every five seconds:

```
```

Figure 15-103. onstat -r Output

Example Output 2: execute `onstat` every ten seconds:

```
```

Figure 15-104. onstat -r 10 Output

Example Output 3: execute `onstat -h` every one second:
Example Output 4: execute ‘onstat -Fh’ every five seconds:
onstat -R: Print LRU, FLRU, and MLRU queue information

Syntax:

```
\> onstat -R
```

Use the -R option to display detailed information about the LRU queues, FLRU queues, and MLRU queues. For an in-depth discussion of the three types of queues, see LRU queues in the shared-memory chapter of the *IBM Informix Administrator’s Guide*.

For each queue, onstat -R lists the number of buffers in the queue and the number and percentage of buffers that have been modified.

Example Output:

```
Figure 15-106. onstat -Rfh Output


Fg Writes    LRU Writes    Chunk Writes
0            0             21

address     flusher     state     data    # LRU    Chunk    Wakeups    Idle Tim
460e6820    0            1            0        0        2        5        9.820

states: Exit Idle Chunk Lru

Buffer pool page size: 2048

buffer hash chain length histogram
# of chains    of len
   6342        0
   1850        1
   8192 total chains
   1850 hashed buffs
   5000 total buffs


Fg Writes    LRU Writes    Chunk Writes
0            0             21

address     flusher     state     data    # LRU    Chunk    Wakeups    Idle Tim
460e6820    0            1            0        0        2        10       22.755

states: Exit Idle Chunk Lru

Buffer pool page size: 2048

buffer hash chain length histogram
# of chains    of len
   4396        0
   3796        1
   8192 total chains
   3796 hashed buffs
   5000 total buffs
```
Output Description:

You can interpret output from this option as follows:

**Buffer pool page size**

Is the page size of the buffer pool in bytes

**#**

Shows the queue number

Each LRU queue is composed of two subqueues: an FLRU queue and a MLRU queue. (For a definition of FLRU and MLRU queues, see LRU queues in the shared-memory chapter of the *IBM Informix Administrator’s Guide.*) Thus, queues 0 and 1 belong to the first LRU queue, queues 2 and 3 belong to the second LRU queue, and so on.

**f/m**

Identifies queue type

This field has four possible values:

- **f** Free LRU queue
  
  In this context, free means not modified. Although nearly all the buffers in an LRU queue are available for use, the
The database server attempts to use buffers from the F LRU queue rather than the M LRU queue. (A modified buffer must be written to disk before the database server can use the buffer.)

F  Free LRU with fewest elements
   The database server uses this estimate to determine where to put unmodified (free) buffers next.

m  MLRU queue

M  MLRU queue that a flusher is cleaning

length  Tracks the length of the queue measured in buffers

% of  Shows the percent of LRU queue that this subqueue composes
   For example, suppose that an LRU queue has 50 buffers, with 30 of those buffers in the MLRU queue and 20 in the F LRU queue. The % of column would list percents of 60.00 and 40.00, respectively.

pair total  Provides the total number of buffers in this LRU queue

priority levels  Displays the priority levels: LOW, MED_LOW, MED_HIGH, HIGH

The -R option also lists the priority levels.

Summary information follows the individual LRU queue information. You can interpret the summary information as follows:

dirty  Is the total number of buffers that have been modified in all LRU queues

queued  Is the total number of buffers in LRU queues

total  Is the total number of buffers

hash buckets  Is the number of hash buckets

buffer size  Is the size of each buffer

start clean  Is the value of LRU_MAX_DIRTY

stop at  Is the value of LRU_MIN_DIRTY

priority downgrades  Is the number of LRU queues downgraded to a lower priority.

priority upgrades  Is the number of LRU queues upgraded to a higher priority.

---

**onstat -s: Print latch information**

**Syntax:**

```bash
$ onstat -s
```

Use the -s option to display general latch information.

**Example Output:**
### Output Description:

You can interpret output from this option as follows:

<table>
<thead>
<tr>
<th>name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Identifies the resource that the latch controls with the following abbreviations:</td>
</tr>
<tr>
<td>archive</td>
<td>Storage-space backup</td>
</tr>
<tr>
<td>bf</td>
<td>Buffers</td>
</tr>
<tr>
<td>bh</td>
<td>Hash buffers</td>
</tr>
<tr>
<td>chunks</td>
<td>Chunk table</td>
</tr>
<tr>
<td>ckpt</td>
<td>Checkpoints</td>
</tr>
<tr>
<td>dbspace</td>
<td>Dbspace table</td>
</tr>
<tr>
<td>flushctl</td>
<td>Page-flusher control</td>
</tr>
<tr>
<td>flushr</td>
<td>Page cleaners</td>
</tr>
<tr>
<td>locks</td>
<td>Lock table</td>
</tr>
<tr>
<td>loglog</td>
<td>Logical log</td>
</tr>
<tr>
<td>LRU</td>
<td>LRU queues</td>
</tr>
<tr>
<td>physb1</td>
<td>First physical-log buffer</td>
</tr>
<tr>
<td>physb2</td>
<td>Second physical-log buffer</td>
</tr>
<tr>
<td>physlog</td>
<td>Physical log</td>
</tr>
<tr>
<td>pt</td>
<td>Tblspace tblspace</td>
</tr>
<tr>
<td>tblsps</td>
<td>Tblspace table</td>
</tr>
<tr>
<td>users</td>
<td>User table</td>
</tr>
</tbody>
</table>

| address | Is the address of the latch                        |

This address appears in the `-u` (users) output wait field if a thread is waiting for the latch.

| lock    | Indicates if the latch is locked and set          |

The codes that indicate the lock status (1 or 0) are computer dependent.

| wait    | Indicates if any user thread is waiting for the latch |

| userthread | Is the shared-memory address of any user thread that is waiting for a latch |

Instead this field contains the thread-control block address, which all threads have. You can compare this address with the user addresses in the `onstat -u` output to obtain the user-process identification number.
To obtain the \texttt{rstcb} address from the \texttt{tcb} address, examine the output of the \texttt{onstat -g ath} option, which lists both addresses for each user thread.

### onstat -t and -T: Print tblspace information

**Syntax:**

\[
\texttt{onstat -t -T}
\]

Use the \texttt{-t} option to display tblspace information for active tblspaces, including whether tblspaces are memory resident. Use the \texttt{-T} option to display the total number of tblspaces.

**Example Output:**

```sql
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 03:58:08 -- 15360 Kbytes

Tblspaces
 n address flgs ucnt tblnum physaddr npages nused npdata nrows nextns
62 a40dc70 0 1 100001 1:14 250 250 0 0 1
195 ac843e0 0 1 1000df 1:236 16 9 4 53 2
2 active, 221 total
```

*Figure 15-109. onstat -t Output*

**Output Description:**

You can interpret output from this option as follows:

- **n**
  - Is a counter of open tblspaces

- **address**
  - Is the address of the tblspace in the shared-memory tblspace table

- **flgs**
  - Uses the following flag bits to describe the flag:
    - 0x00000001: Partition structure is being initialized
    - 0x00000002: Partition was modified. The modified pages have not been flushed to disk.
    - 0x00000004: Partition is being dropped
    - 0x00000008: Partition is for a pseudo table
    - 0x00000010: Partition is being altered in an ADD INDEX or DROP INDEX operation
    - 0x00000020: Partition is being altered in an ALTER TABLE operation
    - 0x00000080: Partition is being dropped while the dbspace is down
    - 0x00000100: Simple large objects in blobspaces are not deleted when the table is dropped
    - 0x00000200: Partition alter page count is updated
Pages have been altered to the latest database schema
System temp table
User temp table
Partition is resident
Index operations are deferred during recovery
Partition is being truncated
Partition is partially truncated

ucnt
Is the usage count, which indicates the number of user threads currently accessing the tblspace

tblnum
Is the tblspace number expressed as a hexadecimal value
The integer equivalent appears as the partnum value in the systables system catalog table.

physaddr
Is the physical address (on disk) of the tblspace

npages
Is the number of pages allocated to the tblspace

nused
Is the number of used pages in the tblspace

npdata
Is the number of data pages used

nrows
Is the number of data rows used

nextns
Is the number of noncontiguous extents allocated
This number is not the same as the number of times that a next extent has been allocated.

resident
Indicates whether tblspace is memory-resident; 1 = yes, 0 = no

The -t option also lists the number of active tblspaces and the total number of tblspaces.

**onstat -u: Print user activity profile**

**Syntax:**

```
onstat -u
```

Use the -u option to print a profile of user activity.

**Example Output:**
Output Description:

The `-u` option provides the following output for each user thread.

*address* Is the shared-memory address of the user thread (in the user table)

Compare this address with the addresses displayed in the `-s` output (latches); the `-b`, `-B`, and `-X` output (buffers); and the `-k` output (locks) to learn what resources this thread is holding or waiting for.

*flags* Provides the status of the session.

The flag codes for position 1:

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>Waiting for a buffer</td>
</tr>
<tr>
<td>C</td>
<td>Waiting for a checkpoint</td>
</tr>
<tr>
<td>G</td>
<td>Waiting for a write of the logical-log buffer</td>
</tr>
<tr>
<td>L</td>
<td>Waiting for a lock</td>
</tr>
<tr>
<td>S</td>
<td>Waiting for mutex</td>
</tr>
<tr>
<td>T</td>
<td>Waiting for a transaction</td>
</tr>
<tr>
<td>Y</td>
<td>Waiting for condition</td>
</tr>
<tr>
<td>X</td>
<td>Waiting for a transaction cleanup (rollback)</td>
</tr>
<tr>
<td>DEFUNCT</td>
<td>The thread has incurred a serious assertion failure, and has been suspended to allow other threads to continue their work.</td>
</tr>
</tbody>
</table>

The flag code for position 2:

* Transaction active during an I/O failure

The flag code for position 3:

A A dbspace backup thread

For other values that appear here, see the third position of flag codes for the `-x` option.

The flag code for position 4:

P Primary thread for a session

The flag codes for position 5:

R Reading
X Thread in critical section

The flag codes for position 7:
B A B-tree cleaner thread
C Terminated user thread waiting for cleanup
D A daemon thread
F A page-cleaner thread
M Special ON-Monitor thread (UNIX)

(sessid) Is the session identification number
During operations such as parallel sorting and parallel index
building, a session might have many user threads associated with
it. For this reason, the session ID identifies each unique session.

(user) Is the user login name (derived from the operating system)

(tty) Indicates the tty that the user is using (derived from the operating
system)
This field is blank on Windows.

(wait) If the user thread is waiting for a specific latch, lock, mutex, or
condition, this field displays the address of the resource. Use this
address to map to information provided in the -s (latch) or -k
(lock) output. If the wait is for a persistent condition, run a grep
for the address in the onstat -a output.

(tout) Is the number of seconds left in the current wait
If the value is 0, the user thread is not waiting for a latch or lock.
If the value is -1, the user thread is in an indefinite wait.

(locks) Is the number of locks that the user thread is holding
(The -k output should include a listing for each lock held.)

(nreads) Is the number of disk reads that the user thread has executed

(nwrites) Is the number of write calls that the user thread has executed
All write calls are writes to the shared-memory buffer cache.

The last line of onstat -u output displays the maximum number of concurrent user
threads that were allocated since you initialized the database server. For example,
the last line of a sample onstat -u output is as follows:
4 active, 128 total, 17 maximum concurrent

The last part of the line, 17 maximum concurrent, indicates that the maximum
number of user threads that were running concurrently since you initialized the
database server is 17.

The output also indicates the number of active users and the maximum number of
users allowed.
onstat -x: Print database server transaction information

Syntax:

```
----onstat- -x----
```

Use the -x option to display transaction information on the database server. The transaction information is required only in the following situations:

- X/Open environment
- Database server participation in distributed queries
- Database server uses the Microsoft® Transaction Server (MTS) transaction manager

Example Output:

```
IBM Informix Dynamic Server Version 11.10.UC1 -- On-Line -- Up 03:58:41 -- 15360 Kbytes

Transactions
address flags userthread locks beginlg curlog logposit isol retrys coord
a509018 A---- a4d8018 0 0 6 0x17304c COMMIT 0
a5091e8 A---- a4d8628 0 0 0 COMMIT 0
a5093e8 A---- a4d8c38 0 0 0 COMMIT 0
a5095e8 A---- a4d9248 0 0 0 COMMIT 0
a509758 A---- a4d9858 0 0 0 COMMIT 0
a509928 A----S a4d9e68 1 0 0 0x0 COMMIT 0 xps_qa
6 active, 128 total, 8 maximum concurrent
```

Output Description:

You can interpret output from onstat -x as follows:

- **address**: Is the shared-memory address of the transaction structure
- **flags**: The flag codes for position 1 (current transaction state):
  - **A**: User thread attached to the transaction
  - **S**: TP/XA suspended transaction
  - **C**: TP/XA waiting for rollback
- **flags**: The flag codes for position 2 (transaction mode):
  - **T**: Tightly-coupled mode (MTS)
  - **L**: Loosely-coupled mode (default mode)
- **flags**: The flag codes for position 3 (transaction stage):
  - **B**: Begin work
  - **P**: Distributed query prepared for commit
  - **X**: TP/XA prepared for commit
  - **C**: Committing or committed
  - **R**: Rolling back or rolled back
Heuristically rolling back or rolled back

The flag code for position 4:
X  XA transaction

The flag codes for position 5 (type of transaction):
G  Global transaction
C  Distributed query coordinator
S  Distributed query subordinate
B  Both distributed query coordinator and subordinate

userthread  Is the thread that owns the transaction (rstm address)
locks     Is the number of locks that the transaction holds
beginlg    Is the log in which the BEGIN WORK record was logged
curlog     Is the current log that the transaction is writing to
logposit   Is the log position

The format of a 4-byte log position is 0xPPPPPBBB, where PPPPP is the page offset in the log and BBB is the byte offset in the page. The logposit can refer to a maximum of 0x100000 (or 1048576) pages in a log file.

For example, a record on the first page of log 12, at a byte offset of 24 would have a log position of 0x18 (page 0, byte offset 18). For more information, see “Determining the Position of a Logical-Log Record” on page 15-143.

isol     Is the isolation level.
retrys   Are the attempts to start a recovery thread for the distributed query
coord    Is the name of the transaction coordinator when the subordinate is executing the transaction

This field tells you which database server is coordinating the two-phase commit.

The last line of the onstat -x output indicates that 8 is the maximum number of concurrent transactions since you initialized the database server.
8 active, 128 total, 8 maximum concurrent

**Determining the Position of a Logical-Log Record**

The curlog and logposit fields provide the exact position of a logical-log record. If a transaction is not rolling back, curlog and logposit describe the position of the most recently written log record. When a transaction is rolling back, these fields describe the position of the most recently “undone” log record. As the transaction rolls back, the curlog and logposit values decrease. In a long transaction, the rate at which the logposit and beginlg values converge can help you estimate how much longer the rollback is going to take.

For an onstat -x example, see monitoring a global transaction in the chapter on multiphase commit protocols in the IBM Informix Administrator’s Guide.
Determining the Mode of a Global Transaction

The onstat -x utility is useful for determining whether a global transaction is executing in loosely-coupled or tightly-coupled mode. The second position of the flags column displays the flags for global transactions. The T flag indicates tightly-coupled mode and the L flag indicates loosely-coupled mode.

Loosely-coupled mode means that the different database servers coordinate transactions but do not share locks. Each branch in a global transaction has a separate transaction XID. The records from all branches display as separate transactions in the logical log.

Tightly-coupled mode means that the different database servers coordinate transactions and share resources such as locking and logging. In a global transaction, all branches that access the same database share the same transaction XID. Log records for branches with the same XID appear under the same session ID. MTS uses tightly-coupled mode.

onstat -X: Print thread information

Syntax:

```

```

Use the -X option to obtain precise information about the threads that are waiting for buffers. For each buffer in use, the -X option displays general buffer information that is also available with either the -b or -B option. For more information, refer to onstat -b in "onstat -b: Print buffer information" on page 15-10.

Example Output:

```

Buffers (Access)
address owner flags pagenum memaddr nslots pgflgs scount waiter
Buffer pool page size: 2048
  0 modified, 3000 total, 4096 hash buckets, 2048 buffer size
Buffer pool page size: 8192
  0 modified, 1000 total, 1024 hash buckets, 8192 buffer size
```

Figure 15-112. onstat -X Output

Output Description:

The onstat -X option has a waiter field to list all user threads that are waiting for the buffer, whereas the onstat -b and -B options contain a waitlist field that displays the address of the first user thread that is waiting for the buffer. The maximum number of shared buffers is specified in the buffers field in the BUFFERPOOL configuration parameter in the ONCONFIG file.

*Buffer pool page size*

is the size of the buffer pool pages in bytes

*address*

Is the address of the buffer header in the buffer table

*flags*

Flags identifying the current status of the buffer page:
0x01  Modified Data
0x02  Data
0x04  LRU
0x08  Error
0x20  LRU AIO write in progress
0x40  Chunk write in progress
0x80  Buffer is/will be result of read-ahead
0x100 Cleaner assigned to LRU
0x200 Buffer should avoid bf_check calls
0x400 Do log flush before writing page
0x800 Buffer has been 'buff' -checked
0x8000 Buffer has been pinned

`pagenum`  Is the physical page number on the disk
`memaddr`  Is the buffer memory address
`nslots`  Is the number of slot-table entries in the page

This field indicates the number of rows (or portions of a row) that are stored on the page.

`pgflgs`  Uses the following values, alone or in combination, to describe the page type:
1  Data page
2  Tblspace page
4  Free-list page
8  Chunk free-list page
9  Remainder data page
b  Partition resident blobpage
c  Blobspace resident blobpage
d  Blob chunk free-list bit page
e  Blob chunk blob map page
10  B-tree node page
20  B-tree root-node page
40  B-tree branch-node page
80  B-tree leaf-node page
100 Logical-log page
200 Last page of logical log
400 Sync page of logical log
800 Physical log
1000 Reserved root page
2000 No physical log required
8000 B-tree leaf with default flags

`scount`  Displays the number of threads that are waiting for the buffer

`waiter`  Lists the addresses of all user threads that are waiting for the buffer
onstat -z: Clear statistics

Syntax:

```
  onstat -z
```

Use the -z option to clear database server statistics, including statistics that relate to Enterprise Replication, and set the profile counts to 0.

If you use the -z option to reset and monitor the count of some fields, be aware that profile counts are incremented for all activity that occurs in any database that the database server manages. Any user can reset the profile counts and thus interfere with monitoring that another user is conducting.

Return Codes on Exit

The onstat utility returns the following codes on exit.

- GLS failures: -1
- Failed to attach shared memory: -1
- Failed to attach shared memory when running 'onstat -': 255
- All other errors detected by onstat: 1
- No errors detected by onstat: 0
- Administration mode: 7
Part 3. Appendixes
Appendix A. Files That the Database Server Uses

This appendix provides brief summaries of the files that you use when you configure and use the database server. It also includes descriptions of files (and one directory) created and used internally by the database server. For many of these files, your only responsibilities are to recognize that those files are legitimate and refrain from deleting them.

Pathnames that appear in the following format indicate files that reside on UNIX: /directory/filename. Pathnames that appear in the following format indicate files that reside on Windows: \directory\filename.

In some cases, environment variables are used to specify the initial pathname of a file. On UNIX, references to environment variables begin with a dollar sign: $INFORMIXDIR. On Windows, references to environment variables begin and end with percent signs: %INFORMIXDIR%.

### Database Server Files

**Table A-1** lists the database server files and the directories in which they reside.

<table>
<thead>
<tr>
<th>Filename</th>
<th>Directory</th>
<th>Purpose</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>af.xxx</td>
<td>Specified by DUMPDIR configuration parameter</td>
<td>Assertion-failure information</td>
<td>By the database server</td>
</tr>
<tr>
<td>ac_msg.log</td>
<td>/tmp, %INFORMIXDIR\etc</td>
<td>archecker message log (for Technical Support)</td>
<td>By the database server</td>
</tr>
<tr>
<td>ac_config.std</td>
<td>$INFORMIXDIR/etc, %INFORMIXDIR\etc</td>
<td>Template for archecker-parameter values</td>
<td>By the database server</td>
</tr>
<tr>
<td>bar_act.log</td>
<td>/tmp, %INFORMIXDIR\etc</td>
<td>ON–Bar activity log</td>
<td>By ON–Bar</td>
</tr>
<tr>
<td>bldutil.process_id</td>
<td>/tmp, \tmp</td>
<td>Error messages about the sysutils database appear in this file</td>
<td>By the database server</td>
</tr>
<tr>
<td>buildsmi.xxx</td>
<td>/tmp, %INFORMIXDIR\etc</td>
<td>Error messages about SMI database</td>
<td>By the database server</td>
</tr>
<tr>
<td>concdr.sh</td>
<td>$INFORMIXDIR /etc/conv, %INFORMIXDIR\etc</td>
<td>Converts the syscdr database to Version 10.0 format</td>
<td>By the database server</td>
</tr>
<tr>
<td>.conf.dbservername</td>
<td></td>
<td>The onsnmp utility uses this file to obtain the database server configuration</td>
<td>By the database server</td>
</tr>
<tr>
<td>core</td>
<td>Directory from which the database server was invoked</td>
<td>Core dump</td>
<td>By the database server</td>
</tr>
<tr>
<td>Emergency boot files (For filenames, see [A-4])</td>
<td>$INFORMIXDIR/etc, %INFORMIXDIR\etc</td>
<td>Used in a cold restore</td>
<td>By ON–Bar</td>
</tr>
<tr>
<td>gcore (UNIX)</td>
<td>Specified by DUMPDIR configuration parameter</td>
<td>Assertion failure information</td>
<td>By the database server</td>
</tr>
</tbody>
</table>
Table A-1. List of Files That the Database Server Uses (continued)

<table>
<thead>
<tr>
<th>Filename</th>
<th>Directory</th>
<th>Purpose</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>illsrra.xx</td>
<td>$INFORMIXDIR/lib, %INFORMIXDIR\lib</td>
<td>Shared libraries for the database server and some utilities</td>
<td>By install procedure</td>
</tr>
<tr>
<td>.informix (UNIX)</td>
<td>User’s home directory</td>
<td>Set personal environment variables</td>
<td>By the user</td>
</tr>
<tr>
<td>informix.rc (UNIX)</td>
<td>$INFORMIXDIR/etc</td>
<td>Set default environment variables for all users</td>
<td>By the database administrator</td>
</tr>
<tr>
<td>INFORMIXTMP</td>
<td>/tmp, \tmp</td>
<td>Temporary directory for internal files</td>
<td>By the database server</td>
</tr>
<tr>
<td>.inf.servicename</td>
<td>/INFORMIXTMP, drive:INFORMIXTMP</td>
<td>Connection information</td>
<td>By the database server</td>
</tr>
<tr>
<td>.infos.dbservername</td>
<td>$INFORMIXDIR/etc, %INFORMIXDIR\etc</td>
<td>Connection information</td>
<td>By the database server</td>
</tr>
<tr>
<td>.infdirs</td>
<td>/INFORMIXTMP, drive:INFORMIXTMP</td>
<td>Database server discovery file that onsnmp uses</td>
<td>By the database server</td>
</tr>
<tr>
<td>InstallServer.log</td>
<td>C: \temp</td>
<td>Database server installation log</td>
<td>By the database server</td>
</tr>
<tr>
<td>ISM catalog</td>
<td>$INFORMIXDIR/ism, %ISMDIR%</td>
<td>Records saved backup objects and storage volumes that IBM Informix Storage Manager (ISM) uses</td>
<td>By ISM</td>
</tr>
<tr>
<td>ISM logs</td>
<td>$INFORMIXDIR/ism/logs, %ISMDIR\logs</td>
<td>Operator alert messages, backend status, additional ISM information</td>
<td>By ISM</td>
</tr>
<tr>
<td>ISMversion</td>
<td>$INFORMIXDIR/ism, %ISMDIR%</td>
<td>ISM version</td>
<td>During installation</td>
</tr>
<tr>
<td>JVM_vpid</td>
<td>Specified by JVPLLOG configuration parameter</td>
<td>Messages that the Java virtual machine generates</td>
<td>By the Java virtual machine</td>
</tr>
<tr>
<td>JVPLLOG</td>
<td>Specified by JVPLLOG configuration parameter</td>
<td>Messages from the Java virtual processor</td>
<td>By the database server</td>
</tr>
<tr>
<td>.jvpprops</td>
<td>Specified by JVPPROFILE configuration parameter</td>
<td>Template for Java VP properties</td>
<td>During installation</td>
</tr>
<tr>
<td>Message log</td>
<td>Specified by MSGPATH configuration parameter</td>
<td>Error messages and status information</td>
<td>By the database server</td>
</tr>
<tr>
<td>The ONCONFIG file</td>
<td>$INFORMIXDIR/etc, %INFORMIXDIR\etc</td>
<td>Configuration information</td>
<td>By the database administrator</td>
</tr>
<tr>
<td>onconfig</td>
<td>$INFORMIXDIR/etc, %INFORMIXDIR\etc</td>
<td>Default ONCONFIG file (optional)</td>
<td>By the database server administrator</td>
</tr>
<tr>
<td>onconfig.std</td>
<td>$INFORMIXDIR/etc</td>
<td>Template for configuration-parameter values</td>
<td>During installation</td>
</tr>
<tr>
<td>onconfig_servername, servernum</td>
<td>$INFORMIXDIR/etc, %INFORMIXDIR\etc</td>
<td>Configuration information for whole-system restores</td>
<td>By the database server</td>
</tr>
<tr>
<td>onsnmp.servername</td>
<td>/tmp, \tmp</td>
<td>Log file that the onsnmp subagent uses</td>
<td>By onsnmp</td>
</tr>
<tr>
<td>onsvapd.log</td>
<td>/tmp, \tmp</td>
<td>Log file for the database server daemon onsvapd</td>
<td>By onsnmp</td>
</tr>
</tbody>
</table>
### Table A-1. List of Files That the Database Server Uses (continued)

<table>
<thead>
<tr>
<th>Filename</th>
<th>Directory</th>
<th>Purpose</th>
<th>Created</th>
</tr>
</thead>
<tbody>
<tr>
<td>revcdr.sh</td>
<td>$INFORMIXDIR/\etc/conv, %INFORMIXDIR%/\etc/conv</td>
<td>Reverts the syscdr database to an earlier format</td>
<td>By the database server</td>
</tr>
<tr>
<td>servicename.exp</td>
<td>/INFORMIXTMP, drive:\INFORMIXTMP</td>
<td>Connection information</td>
<td>By the database server</td>
</tr>
<tr>
<td>servicename.str</td>
<td>/INFORMIXTMP, drive:\INFORMIXTMP</td>
<td>Connection information</td>
<td>By the database server</td>
</tr>
<tr>
<td>shmem.xxx (UNIX)</td>
<td>Specified by DUMPDIR configuration parameter</td>
<td>Assertion-failure information</td>
<td>By the database server</td>
</tr>
<tr>
<td>sm_versions.std</td>
<td>$INFORMIXDIR/etc, %INFORMIXDIR%/\etc</td>
<td>Identifies storage manager in use</td>
<td>During installation</td>
</tr>
<tr>
<td>snmpd.log</td>
<td>/tmp, \tmp</td>
<td>Log file for the SNMP master agent, snmpd</td>
<td>By onsnmp</td>
</tr>
<tr>
<td>sqlhosts (UNIX)</td>
<td>$INFORMIXDIR/etc</td>
<td>Connection information; contained in the registry on Windows</td>
<td>During installation; modified by the database server administrator</td>
</tr>
<tr>
<td>VP.servername.nnx</td>
<td>/INFORMIXTMP, drive:\INFORMIXTMP</td>
<td>Connection information</td>
<td>By the database server</td>
</tr>
<tr>
<td>xbsa.messages</td>
<td>$INFORMIXDIR/ism/applogs, %ISMDIR%/applogs</td>
<td>XBSA library call information</td>
<td>By ISM</td>
</tr>
</tbody>
</table>

### Descriptions of Files

This section provides short descriptions of the files listed in Table A-1.

**af.xxx**

The database server writes information about an assertion failure to the af.xxx file. The file is stored in the directory that the DUMPDIR configuration parameter specifies. For more information, see the information on monitoring for data inconsistency in your *IBM Informix Administrator’s Guide*.

**ac_msg.log**

When you use archecker with ON–Bar to verify a backup, it writes brief status and error messages to the ON–Bar activity log and writes detailed status and error messages to the archecker message log (ac_msg.log). Technical Support uses the archecker message log to diagnose problems with backups and restores.

You specify the location of the archecker message log with the AC_MSGPATH configuration parameter. For more information, see the *IBM Informix Backup and Restore Guide*.

**ac_config.std**

The ac_config.std file contains the default archecker (archive checking) utility parameters. To use the template, copy it into another file, and modify the values. For a comprehensive list of the archecker parameters and how to use archecker with ON–Bar, see the *IBM Informix Backup and Restore Guide*. 
bar_act.log
As ON–Bar backs up and restores data, it writes progress messages, warnings, and error messages to the ON–Bar activity log (bar_act.log). You specify the location of the ON–Bar activity log with the BAR_ACT_LOG configuration parameter. For more information, see the IBM Informix Backup and Restore Guide.

bldutil.process_id
If the database server cannot build the sysutils database, it creates the bldutil.<process_id> file which contains the error messages. The process_id value is the process ID of the bldutil.sh program. To access this output file, specify $(RESFILE).

buildsmi.xxx
If the database server cannot build the sysmaster database, it places a message in the message log that refers you to the buildsmi.xxx file. This file provides information about why the build failed. For information about the sysmaster database, refer to Chapter 2, “The sysmaster Database,” on page 2-1.

concdr.sh
To convert the syscdr database from 7.31, 9.20, 9.21, 9.3 or 9.4 to 10.0 format, run the concdr.sh script on UNIX or the concdr.bat script on Windows. For details, see the IBM Informix Migration Guide.

.conf.dbservername
The .conf.dbservername file is created when you initialize the database server. The onsnmp utility queries this file to find out the configuration status of the database server. Do not delete this file.

The .conf.dbservername file contains information on shared memory and configuration that allows shared-memory clients to connect to the database server when they use utilities such as onstat or onmode.

core
The core file contains a core dump caused by an assertion failure. The database server writes this file to the directory from which the database server was invoked. For more information on monitoring for data inconsistency, see the chapter on consistency checking in the IBM Informix Administrator’s Guide.

Emergency Boot Files for ON–Bar
The ON–Bar emergency boot files contain the information needed to perform a cold restore, and are updated after every backup. For details, see the IBM Informix Backup and Restore Guide.

The filename for the Dynamic Server emergency boot file is ixbar_hostname.servernum.

gcore.xxx (UNIX)
The database server writes information about an assertion failure to the gcore.xxx file. The file is stored in the directory specified by the DUMPDIR configuration parameter. For more information on monitoring for data inconsistency, see the chapter on consistency checking in the IBM Informix Administrator’s Guide.
The **illlsra.xx** files are shared libraries that the database server and some database server utilities use. The shared libraries, if supported on your platform, are installed in `$INFORMIXDIR/lib` or `%INFORMIXDIR%\lib`.

The naming convention of the Informix shared library filename is as follows:

- **ll** — library class (for example, asf or smd)
- **s** — library subclass (d=DSA; s=standard)
- **rr** — major release number (for example, 07 or 08)
- **a** — library version ID (for example, a or b)
- **xx** — shared-library filename extension (for example, so)

---

**UNIX Only**

Symbolic links to these files are automatically created in `/usr/lib` when the products are installed on your computer.

**Important:** The symbolic links to the shared libraries in `/usr/lib` are automatically created by the product installation procedures. However, if your $INFORMIXDIR is not installed using the standard installation method (for example, your $INFORMIXDIR is NFS-mounted from another computer), you or your system administrator might need to create manually the symbolic links of the shared libraries in `/usr/lib`.

---

**~/.informix**

The `~/.informix` file is the private-environment file. Users can create this file and store it in their home directory. The IBM Informix Guide to SQL: Reference discusses the environment-configuration files.

**informix.rc (UNIX)**

The `/informix.rc` file is the environment-configuration file. You can use it to set environment variables for all users of IBM Informix products. The IBM Informix Guide to SQL: Reference discusses the environment-configuration files.

**INFORMIXTMP**

The **INFORMIXTMP** directory is an internal database server directory. During initialization, the database server creates this directory (if it does not exist yet) for storing internal files that must be local and relatively safe from deletion. The **onsnmp** utility uses the files in the **INFORMIXTMP** directory.

**.inf.servicename**

The database server creates the **.inf.servicename** file if any DBSERVERNAME or DBSERVERALIASES uses a shared-memory connection type. The database server removes the file when you take the database server offline. The name of this file is derived from the servicename field of the **sqlhosts** file or registry.
The database server keeps information about client/server connections in this file. You do not use the `.inf.servicename` file directly. You only need to recognize that it is a legitimate file when it appears in the INFORMIXTMP directory.

If this file is accidentally deleted, you must restart the database server.

**.infos.dbservername**

The database server creates the `.infos.dbservername` file when you initialize shared memory and removes the file when you take the database server offline. This file resides in `$INFORMIXDIR/etc` or `%INFORMIXDIR%\etc`. The name of this file is derived from the DBSERVERNAME parameter in the ONCONFIG configuration file.

The `.infos.dbservername` file contains information on shared memory and configuration that allows shared-memory clients to connect to the database server when they use utilities such as `onstat` or `onmode`. Do not delete this file.

**.infxdirs**

The database server maintains an `.infxdirs` file in the INFORMIXTMP directory. This file contains a line for every INFORMIXDIR from which a database server has been launched. If you remove the `.infxdirs` file, `onsnmp` cannot discover any database servers until the next time you restart the database server. Each time you restart the database server, it re-creates the `.infxdirs` file.

**InstallServer.log (Windows)**

The database server creates the `InstallServer.log` during installation.

**ISM Catalog**

ISM creates the ISM catalog during the `ism_start` initialization. The ISM catalog records information about backup and restore save sets and about storage volumes that the storage manager uses. The ISM catalog records are stored in the `mm`, `index`, and `res` files in the `$INFORMIXDIR/ism` or `%ISMDIR%\ism` directory. For more information, see the `IBM Informix Storage Manager Administrator’s Guide`.

**ISM Logs**

ISM creates several logs during ON–Bar backup and restore operations. The message window in the ISM Administrator GUI displays messages from these logs.

<table>
<thead>
<tr>
<th>Log</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>daemon.log</td>
<td>ISM backend status</td>
</tr>
<tr>
<td>messages</td>
<td>Operator alert messages</td>
</tr>
<tr>
<td>summary</td>
<td>Additional ISM information</td>
</tr>
</tbody>
</table>

For more information, see the `IBM Informix Storage Manager Administrator’s Guide`.

**ISMversion**

The `ISMversion` file, which is installed with the database server, identifies the ISM version. Do not edit this file.
**JVM_vpid**

When the 0x10 bit is on for AF CrASH or the AFDEBUG environment variable is on, all the messages that the Java virtual machine generates are logged into the JVM_vpid file, where vpid is the process ID of the Java virtual processor. For more information, see *J/Foundation Developer’s Guide*.

**JVPLOG**

When JVPDEBUG is set to 1, the database server writes tracing messages to the JVPLOG file. You can adjust the tracing level. On UNIX, you can have multiple JVPLOG files, one for each Java virtual processor. On Windows, only one JVPLOG file exists. To obtain the JVP IDs, use the onstat -g glo command. For more information, see *J/Foundation Developer’s Guide*.

**.jvpprops**

The .jvpprops file sets the Java virtual processor properties. Copy the .jvpprops.template to a new file named .jvpprops, and modify the values. For more information, see *J/Foundation Developer’s Guide*.

**Message Log**

The database server writes status and error information to the message-log file. You specify the filename and location of the message log with the MSGPATH configuration parameter. For more information, refer to "MSGPATH" on page 1-65.

**onconfig.std**

The onconfig.std file serves as the template for creating the ONCONFIG configuration file. To use the template, copy it to another file and modify the values.

**Important**: Do not modify or delete onconfig.std. The database server uses values listed in this file when those values are missing from the ONCONFIG file.

For a comprehensive list of the ONCONFIG parameters, see Chapter 1, "Configuration Parameters," on page 1-1.

**The ONCONFIG File**

The current configuration file is the %INFORMIXDIR%/etc/%ONCONFIG% or SINFOMIXDIR/etc/$ONCONFIG file. The database server uses the ONCONFIG file during initialization.

If you start the database server with oninit and do not explicitly set the ONCONFIG environment variable, the database server looks for configuration values in the onconfig.std file. If no onconfig.std file exists, the database server returns the following error message:

**WARNING**: Cannot access configuration file $INFORMIXDIR/etc/$ONCONFIG.

For more information on the order of files where the database server looks for configuration values during initialization, refer the material on initializing the database server in the *IBM Informix Administrator’s Guide*.

For more information on setting up your ONCONFIG file, refer to the materials on installing and configuring the database server in the *IBM Informix Administrator’s Guide*.
onconfig

The onconfig file is an optional file that you create in the $INFORMIXDIR/etc or %INFORMIXDIR%\etc directory. The onconfig file is the default configuration file if the ONCONFIG environment variable is not set. For more information, refer to processing the configuration file in the IBM Informix Administrator’s Guide.

To create the onconfig file, you can copy onconfig.std or one of your customized configuration files. For more information on setting up your ONCONFIG file, refer to installing and configuring the database server in the IBM Informix Administrator’s Guide.

oncfg_servername.servernum

The database server creates the oncfg_servername.servernum file in the $INFORMIXDIR/etc or %INFORMIXDIR%\etc directory when you initialize disk space. The database server updates the file every time that you add or delete a dbspace, a logical-log file, or a chunk. The database server uses the oncfg_servername.servernum file when it salvages logical-log files during a whole-system restore. The database server derives the name of this file from the values of the DBSERVERNAME and SERVERNUM parameters in the ONCONFIG configuration file.

The database server uses the oncfg_servername.servernum files, so do not delete them. For more information, refer to creating the oncfg_servername.servernum file in the IBM Informix Administrator’s Guide and the IBM Informix Backup and Restore Guide.

onsnmp.servername

The onsnmp subagent uses this log file. For more information, see the IBM Informix SNMP Subagent Guide.

This log file is called onsnmp.servername on Dynamic Server.

onsrvapd.log

The onsrvapd daemon uses this log file. For more information, see the IBM Informix SNMP Subagent Guide.

revcdr.sh

To revert the syscdr database from 10.0 to 9.4, 9.3, 7.31, 9.20, or 9.21 format, run the revcdr.sh script on UNIX or the revcdr.bat script on Windows. For details, see the IBM Informix Migration Guide.

shmем.xxx (UNIX)

The database server writes information about an assertion failure to the shmem.xxx file. The file is stored in the directory that the DUMPDIR configuration parameter specifies. For more information on monitoring for data inconsistency, see the chapter on consistency checking in the IBM Informix Administrator’s Guide.

sm_versions.std

The sm_versions.std file is a template for the sm_versions file that you create. The sm_versions file contains a line identifying the current storage-manager version.
The storage manager uses the data in the `sm_versions` file (no `.std` suffix). To update the storage-manager version, edit the `sm_versions` file and then run the `ism_startup` command. For more information, see the *IBM Informix Backup and Restore Guide*.

**snmpd.log**

The SNMP master agent, `snmpdm` uses this log file. For more information, see the *IBM Informix SNMP Subagent Guide*.

**sqlhosts**

- **UNIX Only**

  The `sqlhosts` file is the *connectivity file* on UNIX platforms. It contains information that lets an IBM Informix client connect to an IBM Informix database server. For more information on the `sqlhosts` file, see client/server communications in the *IBM Informix Administrator's Guide*.

  __End of UNIX Only__

- **Windows Only**

  On Windows, the connectivity information is in the `HKEY_LOCAL_MACHINE\SOFTWARE\INFORMIX\SQLHOSTS` key in the Windows registry.

  __End of Windows Only__

**VP.servername.nnx**

The database server creates the `VP.servername.nnx` file, if needed, when you initialize shared memory. The name of this file comes from `DBSERVERNAME` or `DBSERVERALIASES` in the `ONCONFIG` file, the VP number (`nn`), and an internal identifier (`x`).

The database server keeps information about client/server connections in the `VP.servername.nnx` file. You do not use the file directly. You only need to recognize that it is a legitimate file.

If this file is accidentally deleted, you must restart the database server.

**xbsa.messages**

The `xbsa.messages` log contains XBSA library call information. ON–Bar and ISM use XBSA to communicate with each other. Technical Support would use the `xbsa.messages` log to diagnose problems with ON–Bar and ISM communications.
Appendix B. Trapping Errors  

Occasionally, a series of events causes the database server to return unexpected error codes. If you do not have the appropriate diagnostic tools in place when these events occur, it might be difficult for you to determine the cause of these errors. This section discusses the following diagnostic tools:

- **onmode -I**
- **tracepoints**

Collecting Diagnostics using **onmode -I**

To help collect additional diagnostics, you can use **onmode -I** to instruct the database server to perform the diagnostics collection procedures that the *IBM Informix Administrator’s Guide* describes. To use **onmode -I** when you encounter an error number, supply the *iserrno* and an optional session ID. The -I option is just one of many **onmode** options. For more information about **onmode**, see “In This Chapter” on page 11-1.

**Syntax**

```plaintext
onmode -I iserrno [sid]
```

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Key Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-I iserrno</td>
<td>Error number of the error for which you want to collect diagnostic information</td>
<td>None.</td>
</tr>
<tr>
<td>sid</td>
<td>Session ID of the session for which you want to collect diagnostic information</td>
<td>None.</td>
</tr>
</tbody>
</table>

Creating Tracepoints

*Tracepoints* are useful in debugging user-defined routines written in C. You can create a user-defined tracepoint to send special information about the current execution state of a user-defined routine. Each tracepoint has the following parts:

- A **trace class** groups related tracepoints together so that they can be turned on or off at the same time.
  
  You can either use the built-in trace class called **_myErrors** or create your own. To create your own trace class, you insert rows into the **systraceclasses** system catalog table.

- A **trace message** is the text that the database server sends to the tracing-output file.
  
  You can store internationalized trace messages in the **systracemsgs** system catalog table.

- A **tracepoint threshold** determines when the tracepoint executes.
By default, the database server puts all trace messages in the trace-output file in the tmp directory with the following filename:

```
session_num.trc
```

For more information on tracing user-defined routines, see the IBM Informix DataBlade API Programmer’s Guide.
Appendix C. Event Alarms

The database server provides a mechanism for automatically triggering administrative actions based on an event that occurs in the database server environment. This mechanism is the event-alarm feature. Events can be informative (for example, Backup Complete) or can indicate an error condition that requires your attention (for example, Unable to Allocate Memory).

Using ALARMPROGRAM to Capture Events

On UNIX, use the `alarmprogram.sh` and on Windows, use the `alarmprogram.bat` shell script, for handling event alarms and starting automatic log backups. For the setup instructions, see “ALARMPROGRAM” on page 1-14.

To automate logical-log backups only, two ready-made scripts are provided: `log_full.[sh|bat]` and `no_log.[sh|bat]`. Set ALARMPROGRAM to the full pathname of the script. For information, see “ALARMPROGRAM” on page 1-14.

Setting ALRM_ALL_EVENTS

You can set ALRM_ALL_EVENTS to specify whether ALARMPROGRAM runs for all events that are logged in the MSGPATH or only specified noteworthy events (events greater than severity 1).

Writing Your Own Alarm Script

Alternatively, you can write your own shell script, batch file, or binary program that contains the event-alarm parameters. When an event occurs, the database server invokes this executable file and passes it the event-alarm parameters (see Table C-1 on page C-2). For example, your script can use the `class_id` and `class_msg` parameters to take administrative action when a table failure occurs. Set ALARMPROGRAM to the full pathname of this executable file.

Customizing the ALARMPROGRAM scripts

Follow these steps to customize the `alarmprogram.[sh|bat]` script. You can use `alarmprogram.[sh|bat]` instead of `log_full.[sh|bat]` to automate log backups.

To customize the ALARMPROGRAM scripts:
1. Change the value of ADMINMAIL to the email address of the database server administrator.
2. Change the value of PAGERMAIL to the pager service email address.
3. Set the value of the parameter MAILUTILITY with `/usr/bin/mail` for UNIX and `$INFORMIXDIR/bin/ntmail.exe` for Windows.
4. To automatically back up logical logs as they fill, change BACKUP to yes.
   To stop automatic log backups, change BACKUP to any value other than yes.
5. In the ONCONFIG file, set ALARMPROGRAM to the full pathname of `alarmprogram.[sh|bat]`.
6. Restart the database server.
Alarms with a severity of 1 or 2 do not write any messages to the message log nor send email. Alarms with severity of 3 or greater send email to the database administrator. Alarms with severity of 4 and 5 also notify a pager via email.

**Interpreting Error Messages**

Some of the events that the database server reports to the message log cause it to invoke the alarm program. The class messages indicate the events that the database server reports.

The database server reports a nonzero exit code in the message log. In the alarm program, set the EXIT_STATUS variable to 0 for successful completion and to another number for a failure.

For example, if a thread attempts to acquire a lock, but the maximum number of locks that LOCKS specifies has already been reached, the database server writes the following message to the message log:

```
10:37:22 Checkpoint Completed: duration was 0 seconds.
10:51:08 Lock table overflow - user id 30032, rstcb 10132264
10:51:10 Lock table overflow - user id 30032, rstcb 10132264
10:51:12 Checkpoint Completed: duration was 1 seconds.
```

When the database server invokes `alarmprogram.sh` or your alarm program, it generates a message that describes the severity and class of the event. If the severity is greater than 2, the message takes the following format:

Reasonably severe server event:
Severity: 3
Class ID: 21
Class msg: Database server resource overflow: 'Locks'.
Specific msg: Lock table overflow - user id 30032, rstcb 10132264
See Also: # optional message

The following message appears at the end of each e-mailed message:
This e-mail was generated by the server ALARMPROGRAM script on servername because something untoward just happened to eventname.

---

### Event-Alarm Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>severity</td>
<td>Event severity (See Table C-2 for values.)</td>
<td>integer</td>
</tr>
<tr>
<td>class_id</td>
<td>Event class ID (See Table C-3 for values.)</td>
<td>integer</td>
</tr>
<tr>
<td>class_msg</td>
<td>Event class message (See Table C-3 for messages.)</td>
<td>string</td>
</tr>
<tr>
<td>specific_msg</td>
<td>Event specific messages</td>
<td>string</td>
</tr>
<tr>
<td>see_also</td>
<td>Event see-also file</td>
<td>string</td>
</tr>
</tbody>
</table>

### Event Severity

The first parameter passed to the alarm program is the event-severity code. All events reported to the message log have one of the severity codes listed in Table C-2. Message-log events that have severity 1 do not cause the database server to invoke the alarm program unless the ALRM_ALL_EVENTS configuration parameter, supported by Dynamic Server, Version 10.0 or later, is enabled.
Table C-2. Event-Severity Codes

<table>
<thead>
<tr>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Not noteworthy. The event (for example, date change in the message log) is not reported to the alarm program unless ALRM_ALL_EVENTS is enabled.</td>
</tr>
<tr>
<td>2</td>
<td>Information. No error has occurred, but some routine event completed successfully (for example, checkpoint or log backup completed).</td>
</tr>
<tr>
<td>3</td>
<td>Attention. This event does not compromise data or prevent the use of the system; however, it warrants attention (for example, one chunk of a mirrored pair goes down). Sends e-mail to the system administrator.</td>
</tr>
<tr>
<td>4</td>
<td>Emergency. Something unexpected occurred that might compromise data or access to data (assertion failure, or oncheck reports data corrupt). Take action immediately. Pages the system administrator.</td>
</tr>
<tr>
<td>5</td>
<td>Fatal. Something unexpected occurred and caused the database server to fail. Pages the system administrator.</td>
</tr>
</tbody>
</table>

Event Class ID

An event class ID is an integer that the database server substitutes as the second parameter in your alarm program. Each event class ID is associated with one of the events that causes the database server to run your alarm program.

Class Message

A class message is the text of the message that the database server substitutes for the third parameter of your alarm program when an event causes the database server to run your alarm program. The class messages are different for Dynamic Server and Extended Parallel Server.

Specific Messages

The database server substitutes additional information for the fourth parameter of your alarm program. In general, the text of this message is that of the message written to the message log for the event.

See Also Paths

For some events, the database server writes additional information to a file when the event occurs. The pathname in this context refers to the pathname of the file where the database server writes the additional information.

Event Alarms on Dynamic Server

Table C-3 on page C-3 shows the class IDs and class messages for alarms on Dynamic Server. The first column lists the class IDs that identify each alarm and the second column lists the class messages. For more information about setting the ALARMPROGRAM parameter, which controls alarms, see [ALARMPROGRAM](#) on page 1-14.

Table C-3. Event Alarms on Dynamic Server

<table>
<thead>
<tr>
<th>Class ID</th>
<th>Class Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Table failure: 'dbsname:&quot;owner&quot;.tabname'</td>
</tr>
<tr>
<td>2</td>
<td>Index failure: 'dbsname:&quot;owner&quot;.tabname-idxname'</td>
</tr>
<tr>
<td>3</td>
<td>Blob failure: 'dbsname:&quot;owner&quot;.tabname'</td>
</tr>
<tr>
<td>4</td>
<td>Chunk is offline, mirror is active: chunk number</td>
</tr>
</tbody>
</table>
Table C-3. Event Alarms on Dynamic Server (continued)

<table>
<thead>
<tr>
<th>Class ID</th>
<th>Class Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Dbspace is offline: 'dbspace name'</td>
</tr>
<tr>
<td>6</td>
<td>Internal subsystem failure: 'message'</td>
</tr>
<tr>
<td>7</td>
<td>Database server initialization failure</td>
</tr>
<tr>
<td>8</td>
<td>Physical restore failure</td>
</tr>
<tr>
<td>9</td>
<td>Physical recovery failure</td>
</tr>
<tr>
<td>10</td>
<td>Logical recovery failure</td>
</tr>
<tr>
<td>11</td>
<td>Cannot open chunk: 'pathname'</td>
</tr>
<tr>
<td>12</td>
<td>Cannot open dbspace: 'dbspace name'</td>
</tr>
<tr>
<td>13</td>
<td>Performance improvement possible</td>
</tr>
<tr>
<td>14</td>
<td>Database failure. 'database name'</td>
</tr>
<tr>
<td>15</td>
<td>High-Availability Data-Replication failure</td>
</tr>
<tr>
<td>16</td>
<td>Backup completed: 'dbspace list'</td>
</tr>
<tr>
<td>17</td>
<td>Backup aborted: 'dbspace list'</td>
</tr>
<tr>
<td>18</td>
<td>Log backup completed: log number</td>
</tr>
<tr>
<td>19</td>
<td>Log backup aborted: log number</td>
</tr>
<tr>
<td>20</td>
<td>Logical logs are full—backup is needed</td>
</tr>
<tr>
<td>21</td>
<td>Database server resource overflow: 'resource name'</td>
</tr>
<tr>
<td>22</td>
<td>Long transaction detected</td>
</tr>
<tr>
<td>23</td>
<td>Logical log 'number' complete</td>
</tr>
<tr>
<td>24</td>
<td>Unable to allocate memory</td>
</tr>
<tr>
<td>25</td>
<td>Internal subsystem initialized: 'message' (starts the optical subsystem)</td>
</tr>
<tr>
<td>26</td>
<td>Dynamically added log file logid</td>
</tr>
<tr>
<td>27</td>
<td>Log file required</td>
</tr>
<tr>
<td>28</td>
<td>No space for log file</td>
</tr>
<tr>
<td>29</td>
<td>Chunk (storage) failure</td>
</tr>
<tr>
<td>29</td>
<td>Data capacity</td>
</tr>
<tr>
<td>29</td>
<td>Logical log capacity</td>
</tr>
<tr>
<td>29</td>
<td>Maximum locks</td>
</tr>
<tr>
<td>29</td>
<td>Maximum capacity</td>
</tr>
<tr>
<td>29</td>
<td>Maximum sessions</td>
</tr>
</tbody>
</table>
Appendix D. Discontinued Configuration Parameters

This section lists the discontinued and obsolete configuration parameters for Dynamic Server.

Table D-1 summarizes the discontinued parameters. Although these parameters are still supported, it is recommended that you do not use them. Remove these parameters from the ONCONFIG file before using the VPCLASS parameter.

Table D-1. Discontinued Configuration Parameters

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFF_NPROCS</td>
<td>page D-1</td>
</tr>
<tr>
<td>AFF_SPROC</td>
<td>page D-2</td>
</tr>
<tr>
<td>BUFFERS</td>
<td>page D-2</td>
</tr>
<tr>
<td>FAST_RESTART_CKPT_FUZZYLOG</td>
<td>page D-3</td>
</tr>
<tr>
<td>FAST_RESTART_PHYSLOG</td>
<td>page D-4</td>
</tr>
<tr>
<td>LRU_MAX_DIRTY</td>
<td>page D-5</td>
</tr>
<tr>
<td>LRU_MIN_DIRTY</td>
<td>page D-5</td>
</tr>
<tr>
<td>LRUS</td>
<td>page D-6</td>
</tr>
<tr>
<td>NOAGE</td>
<td>page D-7</td>
</tr>
<tr>
<td>NUMAIIOVPS</td>
<td>page D-7</td>
</tr>
<tr>
<td>NUMCPUVPS</td>
<td>page D-8</td>
</tr>
</tbody>
</table>

Table D-2 summarizes the configuration parameters that are no longer supported.

Table D-2. Obsolete Configuration Parameters

<table>
<thead>
<tr>
<th>Configuration Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>LBU_PRESERVE</td>
<td>page D-5</td>
</tr>
<tr>
<td>LOGSMAX</td>
<td>page D-5</td>
</tr>
</tbody>
</table>

**AFF_NPROCS (Discontinued)**

- **onconfig.std value:** 0
- **units:** Number of CPUs
- **range of values:** 0 through number of CPUs in the computer
- **takes effect:** When the database server shuts down and restarts
- **refer to**
  - Virtual-processor classes, in the chapter on virtual processors and threads in the *IBM Informix Administrator’s Guide*
  - “AFF_SPROC” on page D-2
  - “VPCLASS” on page 1-103
On multiprocessor computers that support processor affinity, AFF_NPROCS specifies the number of CPUs to which the database server can bind CPU virtual processors. Binding a CPU virtual processor to a CPU causes the virtual processor to run exclusively on that CPU. The database server assigns CPU virtual processors to CPUs in serial fashion, starting with the processor number that AFF_SPROC specifies.

If you specify more CPU virtual processors than there are processors, the database server starts over again at the beginning. For example, if you set AFF_NPROCS to 3 and AFF_SPROC to 5, the database server assigns two CPU virtual processors to processor 5, two CPU virtual processors to processor 6, and one CPU virtual processor to processor 7.

**Important:** Use VPCLASS instead of AFF_NPROCS to specify the number of CPUs. You cannot use both AFF_NPROCS and VPCLASS cpu in the same ONCONFIG file.

### AFF_SPROC

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>CPU number</td>
</tr>
<tr>
<td>range of values</td>
<td>0 through (AFF_NPROCS - NUMCPUVPS + 1)</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server shuts down and restarts</td>
</tr>
<tr>
<td>refer to</td>
<td>• Virtual-processor classes, in the chapter on virtual processors and threads in the IBM Informix Administrator’s Guide</td>
</tr>
<tr>
<td></td>
<td>• “AFF_NPROCS (Discontinued)” on page D-1</td>
</tr>
<tr>
<td></td>
<td>• “VPCLASS” on page 1-103</td>
</tr>
</tbody>
</table>

On multiprocessor computers that support processor affinity, AFF_SPROC specifies the CPU, starting with 0, on which the database server starts binding CPU virtual processors to CPUs. The AFF_NPROCS parameter specifies the number of CPUs that the database server will use. The NUMCPUVPS parameter specifies the number of CPU virtual processors to be started, and the AFF_SPROC parameter specifies the CPU on which the first virtual processor is to start. For example, if you assign eight CPUs (AFF_NPROCS = 8), and set NUMCPUVPS to 3 and AFF_SPROC to 5, the database server binds CPU virtual processors to the fifth, sixth, and seventh CPUs.

**Important:** Use VPCLASS instead of AFF_SPROC to specify processor affinity. You cannot use both AFF_SPROC and VPCLASS cpu in the same ONCONFIG file.

### BUFFERS

| onconfig.std value | UNIX: 5000  |
|--------------------| Windows: 2000 |
| units              | Number of buffers |
| range of values    | For 32-bit platform on UNIX: |
with page size equal to 2048 bytes:
100 through 1,843,200 buffers
(1843200 = 1800 * 1024)

with page size equal to 4096 bytes:
100 through 921,600 buffers
(921,600 = ((1800 * 1024) / 4096) * 2048)

For 32-bit platform on Windows:
100 through 524,288 buffers
(524,288 = 512 * 1024)

For 64-bit platforms: 100 through 2^31 - 1 buffers
(For the actual value for your 64-bit platform, see your machine notes. The maximum number of buffers on Solaris is 536,870,912.)

Note: Information that was specified with the BUFFERS configuration parameter prior to Version 10.0 is now specified using the BUFFERPOOL configuration parameter. For more information, see "BUFFERPOOL" on page 1-18.

BUFFERS specifies the maximum number of shared-memory buffers that the database server user threads have available for disk I/O on behalf of client applications. Therefore, the number of buffers that the database server requires depends on the applications. For example, if the database server accesses 15 percent of the application data 90 percent of the time, you need to allocate enough buffers to hold that 15 percent. Increasing the number of buffers can improve system performance.

In general, buffer space should range from 20 to 25 percent of physical memory. It is recommended that you calculate all other shared-memory parameters after you set buffer space (BUFFERS * system_page_size) to 20 percent of physical memory.

---

**FAST_RESTART_CKPT_FUZZYLOG**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>The FAST_RESTART_CKPT_FUZZYLOG parameter does not need to be in the onconfig.std file.</th>
</tr>
</thead>
<tbody>
<tr>
<td>range of values</td>
<td>0 (default) = Disable the flushing of dirty fuzzy pages to the physical log at checkpoint.</td>
</tr>
<tr>
<td></td>
<td>1 = Enable the flushing of dirty fuzzy pages to the physical log at checkpoint.</td>
</tr>
<tr>
<td>takes effect</td>
<td>At the checkpoint that occurs after the parameter is enabled. If the total number of unflushed, dirty fuzzy pages exceeds 20 percent of the total physical</td>
</tr>
</tbody>
</table>

**Note:** Information that was specified with the FAST_RESTART_CKPT_FUZZYLOG parameter does not need to be in the onconfig.std file. For more information, see "FAST_RESTART_CKPT_FUZZYLOG" on page D-3.
log space, the pages will not be written to the physical log. If server fails after this checkpoint, crash recovery receives no performance benefit.

Refer to Information on fast recovery and alternative fast restart recovery options for fuzzy operations in the IBM Informix Administrator’s Guide.

The FAST_RESTART_CKPT_FUZZYLOG parameter and the FAST_RESTART_PHYSLOG parameter enable the database server to perform physical logging on fuzzy checkpoints during the roll-forward (log replay) phase of recovery, thus decreasing recovery time. You can use either parameter or both when using fuzzy checkpoints.

The database server must be online when you enable the FAST_RESTART_CKPT_FUZZYLOG parameter.

**FAST_RESTART_PHYSLOG**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>The FAST_RESTART_PHYSLOG parameter does not need to be in the onconfig.std file.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>range of values</strong></td>
<td>0 (default) = Disable physical logging on fuzzy checkpoints during the roll-forward (log replay) phase of recovery. 1 = Enable physical logging on fuzzy checkpoints during the roll-forward (log replay) phase of recovery, thus decreasing recovery time.</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>Immediately. If the total number of unflushed, fuzzy dirty pages exceeds 20 percent of the total physical log space, the pages will not be written to the physical log. However, if the database server fails before the next checkpoint performs, maximum fast-recovery performance does not occur because the database server did not log all of the fuzzy updates in the checkpoint intervals.</td>
</tr>
<tr>
<td><strong>refer to</strong></td>
<td>Information on fast recovery and alternative fast restart recovery options for fuzzy operations in the IBM Informix Administrator’s Guide</td>
</tr>
</tbody>
</table>

The FAST_RESTART_PHYSLOG parameter and the FAST_RESTART_CKPT_FUZZYLOG parameter enable the database server to perform physical logging on fuzzy checkpoints during the roll-forward (log replay) phase of recovery, thus decreasing recovery time. You can use either parameter or both when using fuzzy checkpoints.

Only use the FAST_RESTART_PHYSLOG parameter if the buffer pool is at least 25 percent larger than the physical buffer size. The buffer pool must be large enough to hold the physical log, log pages, and other pages read during recovery. If the buffer pool is not configured correctly, fast recovery performance is compromised.

The extra physical logging that occurs when the database server uses the FAST_RESTART_PHYSLOG parameter affects runtime performance. If you do not
want to sacrifice runtime performance or if you do not want to increase the buffer size, use the FAST_RESTART_CKPT_FUZZYLOG parameter to reduce some recovery time.

After enabling the FAST_RESTART_PHYSLOG parameter by setting it to 1, you can initiate fast recovery using the oninit utility. Simply execute oninit without any options.

The database server must be online when you enable the FAST_RESTART_PHYSLOG parameter.

**LBU_PRESERVE**

Dynamic Server no longer supports the LBU_PRESERVE parameter, which reserves the last logical log for ON–Archive use. ON–Archive, which has been discontinued, was the only utility that required free log space to back up a logical log.

**LOGSMAX**

Dynamic Server no longer supports the LOGSMAX parameter.

LOGSMAX specifies the maximum number of logical-log files for a database server instance. The database server requires at least three logical-log files for operation. The maximum number of logical logs is 32,767. The LOGSMAX value must be equal to or less than the highest log file number.

**LRU_MAX_DIRTY**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>60.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>units</td>
<td>Percent</td>
</tr>
<tr>
<td>range of values</td>
<td>0 through 100 (fractional values are allowed)</td>
</tr>
<tr>
<td>takes effect</td>
<td>When the database server is shut down and restarted</td>
</tr>
<tr>
<td>refer to</td>
<td>The following topics in the shared-memory chapter of the <em>IBM Informix Dynamic Server Administrator’s Guide</em></td>
</tr>
<tr>
<td></td>
<td>• LRU queues</td>
</tr>
<tr>
<td></td>
<td>• Limiting the number of pages added to the MLRU queues</td>
</tr>
</tbody>
</table>

**Note:** Information that was specified with the LRU_MAX_DIRTY configuration parameter prior to Version 10.0 is now specified using the BUFFERPOOL configuration parameter. For more information, see “BUFFERPOOL” on page 1-18.

LRU_MAX_DIRTY specifies the percentage of modified pages in the LRU queues at which the queue is cleaned. If a parameter is specified out of the range of values, then the default of 60.00 percent is set.

**LRU_MIN_DIRTY**

<table>
<thead>
<tr>
<th>onconfig.std value</th>
<th>50.00</th>
</tr>
</thead>
</table>
LRUS

**onconfig.std value** 8

*if not present*  
If MULTIPROCESSOR is set: MAX(4, num_cpu_vps)  
If MULTIPROCESSOR is not set: 4

**units** Number of LRU queues

**range of values** 1 through 128

**takes effect** When the database server is shut down and restarted

**utilities** `onstat -R` (see [15-134])

**refer to**  
- LRU queues, in the shared-memory chapter of the *IBM Informix Dynamic Server Administrator’s Guide*
- Chapter on configuration effects on memory, in your *IBM Informix Dynamic Server Performance Guide*

**Note:** Information that was specified with the LRU configuration parameter prior to Version 10.0 is now specified using the **BUFFERPOOL** configuration parameter. For more information, see [“BUFFERPOOL” on page 1-18](#).

LRUS specifies the number of LRU (least-recently-used) queues in the shared-memory buffer pool. You can tune the value of LRUS, in combination with the LRU_MIN_DIRTY and LRU_MAX_DIRTY parameters, to control how frequently the shared-memory buffers are flushed to disk.

Setting LRUS too high might result in excessive page-cleaner activity.
NOAGE

**onconfig.std value**

0

**range of values**

0 = Use priority aging,
1 = Disable priority aging.

**takes effect**

When the database server shuts down and restarts

**refer to**

- Preventing priority aging, in the chapter on virtual processors and threads in the IBM Informix Administrator’s Guide
- “VPCLASS” on page 1-103

Some operating systems lower the priority of processes as the processes run over a long period of time. NOAGE, when set to 1, disables priority aging of CPU virtual processors by the operating system. When NOAGE is set to the default of 0, the operating system might lower the priority of CPU virtual processors, as well as other processes, as they accumulate processing time. If your operating system supports priority aging, it is recommended that you set NOAGE to 1.

**Important:** It is recommended that you specify priority aging with the VPCLASS parameter instead of the NOAGE parameter. You cannot use both NOAGE and VPCLASS cpu in the same ONCONFIG file.

NUMAIOVPS

**onconfig.std value**

None

**if not present**

(2 * number_of_chunks) or 6, whichever is greater;

**number_of_chunks** is the number of chunks that you have allocated.

**units**

Number of AIO VPs

**range of values**

Integer greater than or equal to 1

**takes effect**

When the database server shuts down and restarts

**utilities**

onmode -p in onmode -p: Add or remove virtual processors” on page 11-17

**refer to**

- Asynchronous I/O, in the chapter on virtual processors and threads in the IBM Informix Administrator’s Guide
- “VPCLASS” on page 1-103

NUMAIOVPS specifies the number of virtual processors of the AIO class to run. Unless kernel asynchronous I/O is implemented, the AIO virtual processors perform all the database server disk I/O, other than I/O to the log files.

**Important:** It is recommended that you specify the number of AIO VPs with VPCLASS aio instead of NUMAIOVPS. You cannot use both
NUMAIOVPS and VPCLASS aio in the same ONCONFIG file.

**UNIX Only**

If your platform has kernel-asynchronous I/O (KAIO) turned on, the database server uses AIO virtual processors to perform I/O only to cooked chunks. The database server uses KAIO to perform all I/O to raw disk space and to the physical and logical logs. For details, see the machine notes.

End of UNIX Only

---

**NUMCPUVPS**

<table>
<thead>
<tr>
<th><strong>onconfig.std value</strong></th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>units</strong></td>
<td>Number of CPU VPs</td>
</tr>
<tr>
<td><strong>range of values</strong></td>
<td>1 through the number of CPUs</td>
</tr>
<tr>
<td><strong>takes effect</strong></td>
<td>When the database server shuts down and restarts</td>
</tr>
<tr>
<td><strong>utilities</strong></td>
<td>onmode -p in <a href="#">“onmode -p: Add or remove virtual processors” on page 11-17</a></td>
</tr>
</tbody>
</table>
| **refer to**           | • CPU virtual processors, in the chapter on virtual processors and threads in the IBM Informix Administrator’s Guide  
                          • "VPCLASS” on page 1-103 |

NUMCPUVPS specifies the number of virtual processors of the CPU class to run. CPU virtual processors run all threads that start as the result of a connection by a client application, as well as internal threads. In general, allocate only one CPU virtual processor on a single-processor computer or node. On a multiprocessor computer or node, do not allocate more CPU virtual processors than there are CPUs.

**Important:** It is recommended that you specify the number of CPU virtual processors with VPCLASS cpu instead of NUMCPUVPS. You cannot use both NUMCPUVPS and VPCLASS cpu in the same ONCONFIG file.

On UNIX, use the onmode -p -1 CPU command to decrease the number of CPU VPs. On Windows, you can add a CPU VP, but you cannot subtract it.
Appendix E. Error Messages

This chapter lists nonnumbered messages that are printed in the database server message log and provides corrective actions.

For information on numbered messages and the unnumbered ON–Bar messages, search for the message text in the error messages file, which is located in the subdirectory for your locale under the $INFORMIXDIR/msg directory. You can also search IBM Informix Error Messages in English at the IBM Informix Online Documentation site at [http://www.ibm.com/software/data/informix/pubs/](http://www.ibm.com/software/data/informix/pubs/)

Some of the messages included below might require you to contact Technical Support staff. Such messages are rarely, if ever, seen at customer locations.

For information on what the message log is, see installing and configuring the database server in the IBM Informix Administrator’s Guide. For information on specifying the path to the message file, see “MSGPATH” on page 1-65

How the Messages Are Ordered in This Chapter

Database server message-log messages are arranged in this chapter in alphabetical order, sorted with the following additional rules:

• The time stamp that precedes each message is ignored.
• Letter case is ignored in alphabetization.
• Spaces are ignored.
• Quotation marks are ignored.
• Leading ellipses are ignored.
• The word the is ignored if it is the first word in the message.
• Messages that begin with numbers or punctuation symbols appear toward the end of the list in a special section labeled “Messages: Symbols” on page E-20
• Certain related messages are grouped together, as follows:
  – “Conversion/Reversion Messages” on page E-21
  – “Conversion and Reversion Messages for Enterprise Replication” on page E-26
  – “Dynamic Log Messages” on page E-27
  – “Sbspace Metadata Messages” on page E-28
  – “Truncate Table Messages” on page E-29

A cause and suggested corrective action for a message or group of messages follow the message text.

How to View These Messages

Use one of the following methods to view these messages:

• Online message log
  To see the messages displayed as they occur, use the tail -f online.log command.
• onstat -m command
For more information, see "onstat -l: Print physical and logical log information" on page 15-122.

- IBM Informix Server Administrator (ISA)
  For more information, see the ISA online help.

To see the error number associated with these unnumbered messages, view the logmessage table in the sysmaster database:

```
SELECT * FROM logmessage;
```

**Message Categories**

Four general categories of unnumbered messages exist, although some messages fall into more than one category:

- Routine information
- Assertion-failed messages
- Administrative action needed
- Unrecoverable error detected

Technical Support uses the assertion-failed messages to assist in troubleshooting and diagnostics. The information that they report often falls into the category of *unexpected events* that might or might not develop into problems caught by other error codes. Moreover, the messages are terse and often extremely technical. They might report on one or two isolated statistics without providing an overall picture of what is happening. This information can suggest to technical support possible research paths.

**Messages: A-B**

---

**Aborting Long Transaction: tx 0xn.**

**Cause:** The transaction spans the log space specified by transaction high-watermark (LTXHWM), and the offending long transaction is rolling back.

**Action:** No additional action is needed. The address of the transaction structure in shared memory is displayed as a hexadecimal value.

---

**Affinitied VP nm to phys proc nn.**

**Cause:** The database server successfully bound a CPU virtual processor to a physical processor.

**Action:** None required.

---

**Affinity not enabled for this server.**

**Cause:** You tried to bind your CPU virtual processors to physical processors, but the database server that you are running does not support process affinity.

**Action:** Set AFF_NPROCS to 0, or remove the affinity setting from VPCLASS.

---

**Assert Failed: Error from SBSpace cleanup thread.**

**Cause:** The sbspace cleanup thread encountered an error while cleaning up stray smart large objects.

**Action:** See the action suggested in the message log file.

Most of the time, running onspaces -cl sbspacename on the failed sbspace succeeds in cleaning up any stray smart large objects. If you encounter an unrecoverable error, contact Technical Support.

**Assert Failed: Short description of what failed**

**Who:** Description of user/session/thread running at the time

**Result:** State of the affected database server entity

**Action:** What action the database administrator should take

See Also: DUMPDIR/af.uniqid containing more diagnostics.

**Cause:** This message indicates an internal error.

**Action:** The af.uniqid file in the directory specified by the ONCONFIG parameter DUMPDIR contains a copy of the assertion-failure message that was sent to the message log, as well as the contents of the current, relevant structures and/or data buffers. The information included in this message is intended for Technical Support.

**Begin re-creating indexes deferred during recovery.**
Cause: During recovery, indexes to be created are deferred until after recovery completes. This message indicates that the database server deferred re-creating indexes and that it is now creating the indexes. During the time that the database server re-creates the indexes, it locks the affected tables with a shared lock.

Action: None required.

Building 'sysmaster' database requires mm pages of logical log. Currently there are m pages available. Prepare to back up your logs soon.

Messages: C

Cannot Allocate Physical-log File, mm wanted, m available.

Cause: The database server attempted to initialize shared memory with a physical-log size that exceeds the amount of contiguous space available in the dbspace (specified as PHYSDBS in ONCONFIG). Both quantities of space, wanted and available, are expressed as kilobytes.

Action: You must either reduce the size of the physical log (specified as PHYSFILE in ONCONFIG) or change the location of the physical log to a dbspace that contains adequate contiguous space to accommodate the physical log.

Cannot create a user-defined VP class with 'SINGLE_CPU_VP' non-zero.

Cause: SINGLE_CPU_VP is set to nonzero, and onmode was used to create a user-defined VP class.

Action: If user-defined VP classes are necessary, stop the database server, change SINGLE_CPU_VP to zero, and restart the database server.

Cannot alter a table which has associated violations table.

Cause: The user tried to add, drop, or modify a column in a table that has a violations table associated with it.

Action: Do not change the columns in the user table.

Cannot change to mode.

Cause: Some error during fast or full recovery has prevented the system from changing to online or quiescent mode.

Action: See previous messages in the log file for information.

Cannot create violations/diagnostics table.

Cause: The user issued a START VIOLATIONS TABLE statement for a target table. The database server cannot create the violations table for this target table. Any of the following situations might be the reason for this failure:

- The target table already has a violations table.
- You specified an invalid name for the violations table in the START VIOLATIONS TABLE statement. For example, if you omit the USING clause from the statement and if the number of characters in the target table plus four characters is longer than the maximum identifier length, the generated names of the violations table exceed the maximum identifier length.
- You specified a name for the violations table in the START VIOLATIONS TABLE statement that match the names of existing tables in the database.
- The target table contains columns with the names informix_tupleid, informix_optype, or informix_recowner. Because these column names duplicate the informix_tupleid, informix_optype, or informix_recowner columns in the violations table, the database server cannot create the violations table.
- The target table is a temporary table.
- The target table is serving as a violations table for some other table.
- The target table is a system catalog table.

Action: To determine if you need to take action, examine the logical log as described in "Interpreting Logical-Log Records," on page 5-1.
**Action:** To resolve this error, perform one of the following actions:

- If the violations table name was invalid, specify a unique name for the violations table in the USING clause of the START VIOLATIONS TABLE statement.
- If the target table contains columns with the names informix_tupleid, informix_opatype, or informix_recowner, rename them to something else.
- Choose a permanent target table that is not a system catalog table or a violations table for some other table.

---

**Cannot insert from the violations table to the target table.**

**Cause:** The user has issued a statement that attempts to insert rows from the violations table into the target table. For example, the user enters the following invalid statement:

```
INSERT INTO mytable SELECT * FROM mytable_vio;
```

Also, if the target table has filtering-mode constraints, you receive this error. Extended Parallel Server does not support filtering-mode constraints.

**Action:** To recover from this error, perform the following actions:

- Do not use filtering constraints.
- Stop the violations table.
- Insert rows from the violations table into a temporary table, and then insert rows from the temporary table into the target table.

---

**Cannot modify/drop a violations/diagnostics table.**

**Cause:** The user has tried to alter or drop a table that is serving as a violations table for another table.

**Action:** Do not alter or drop the violations table.

---

**Cannot Open Dbspace nnn.**

**Cause:** The database server is unable to access the specified dbspace. This message indicates a problem opening the tblspace or corruption in the initial chunk of the dbspace.

**Action:** Verify that the device or devices that make up the chunks of this dbspace are functioning properly and that you assigned them the correct operating-system permissions (rw-rw----). You might be required to perform a data restore.

---

**Cannot Open Logical Log.**

**Cause:** The database server is unable to access the logical-log files. Because the database server cannot operate without access to the logical log, you must resolve this problem.

**Action:** Verify that the chunk device where the logical-log files reside is functioning and has the correct operating-system permissions (rw-rw----).

---

**Cannot Open Mirror Chunk pathname, errno = nnn.**

**Cause:** The database server cannot open the mirrored chunk of a mirrored pair. The chunk pathname and the operating-system error are returned.

**Action:** For more information about corrective actions, see your operating-system documentation.

---

**Cannot Open Primary Chunk chunkname.**

**Cause:** The initial chunk of a mirrored pair cannot be opened. The chunk pathname and the operating-system error are returned.

**Action:** For more information about corrective actions, see your operating-system documentation.

---

**Cannot open sysams in database name, iserrno number.**

**Cause:** An error occurred when the database server opened the sysams system table.

**Action:** Note the error number and contact Technical Support.

---

**Cannot open sysdistrib in database name, iserrno number.**

**Cause:** An error occurred when the database server accessed the sysdistrib system table.

**Action:** Note the error number and contact Technical Support.

---

**Cannot open system_table in database name, iserrno number.**

**Cause:** An error occurred when the database server opened the specified system table.

**Action:** Note the error number and contact Technical Support.

---

**Cannot open systrigbody in database name, iserrno number.**

**Cause:** An error occurred when the database server accessed the systrigbody system table.

**Action:** Note the error number and contact Technical Support.
Cannot open systriggers in database name, Iserrno number.

Cause: An error occurred when the database server accessed the systriggers system table.

Action: Note the error number and contact Technical Support.

Cannot open sysxtdtypes in database name, Iserrno number.

Cause: An error occurred while accessing the sysxtdtypes system table.

Action: Note the error number and contact Technical Support.

Cannot Perform Checkpoint, shut system down.

Cause: A thread that is attempting to restore a mirrored chunk has requested a checkpoint, but the checkpoint cannot be performed.

Action: Shut down the database server.

Cannot Restore to Checkpoint.

Cause: The database server is unable to recover the physical log and thus unable to perform fast recovery.

Action: If the database server does not come online, perform a data restore from dbspace backup.

Cannot Rollback Incomplete Transactions.

Cause: Within the fast-recovery or data-restore procedure, the logical-log records are first rolled forward. Then, open transactions that have not committed are rolled back. An open transaction could fail during the rollback, leaving some of the modifications from the open transaction in place. This error does not prevent the database server from moving to quiescent or online mode, but it might indicate an inconsistent database.

Action: To determine if any action is needed, use the onlog utility to examine the logical log.

Cannot update pagezero.

Cause: A failure occurred while the database server was trying to rewrite a reserved page during the reversion process.

Action: See previous messages in the log file for information, or contact Technical Support.

Cannot update syscasts in database name, Iserrno number.

Cause: An internal error occurred while inserting data into the syscasts system table.

Action: Contact Technical Support.

Can't affinity VP mmm to phys proc nn.

Cause: The database server supports process affinity, but the system call to bind the virtual processor to a physical processor failed.

Action: See your operating-system documentation.

Changing the sbspace minimum extent value: old value value1, new value value2.

Cause: This informational message occurs when you issue the following command:

onspaces -ch sbspace -0f "MIN_EXT_SIZE=value1" -y

Action: None. For more information, see onspaces "ch: Change sbspace default specifications" on page 14-17

Checkpoint blocked by down space, waiting for override or shutdown.

Cause: A dbspace has gone down during a checkpoint interval. The database server is configured to wait for an override when this situation occurs.

Action: Either shut down the database server or issue an onmode -O command to override the down dbspace. For more information on the onmode utility, see "In This Chapter" on page 11-1

Checkpoint Completed: duration was n seconds.

Cause: A checkpoint completed successfully.

Action: None required.

Checkpoint Page Write Error.

Cause: The database server detected an error in an attempt to write checkpoint information to disk.

Action: For additional assistance in resolving this situation, contact Technical Support.

Checkpoint Record Not Found in Logical Log.

Cause: The logical log or the chunk that contains the logical log is corrupted. The database server cannot initialize.

Action: Perform a data restore from dbspace backup.
Chunk chunkname added to space spacename.

**Cause:** The variables in this message have the following values:

- **chunkname** is the name of the chunk that the database server administrator is adding.
- **spacename** is the name of the storage space to which the database server administrator is adding the chunk.

**Action:** None required.

---

Chunk chunkname dropped from space spacename.

**Cause:** The database server administrator dropped chunk chunkname from space spacename.

**Action:** None required.

---

Chunk number nn pathname -- Offline.

**Cause:** The indicated chunk in a mirrored pair has been marked with status 0 and taken offline. The other chunk in the mirrored pair is operating successfully.

**Action:** Take steps now to repair the chunk device and restore the chunk. The chunk number and chunk device pathname are displayed.

---

Chunk number nn pathname -- Online.

**Cause:** The indicated chunk in a mirrored pair has been recovered and is online (marked with status 0). The chunk number and chunk device pathname are displayed.

**Action:** None required.

---

The chunk pathname must have READ/WRITE permissions for owner and group.

**Cause:** The chunk pathname does not have the correct owner and group permissions.

**Action:** Make sure that you assigned the correct permissions (-rw-rw--) to the device on which the chunk is located.

---

The chunk pathname must have owner-ID and group-ID set to informix.

**Cause:** The chunk chunkname does not have the correct owner and group ID.

**Action:** Make sure the device on which the chunk is located has the ownership. On UNIX, both owner and group should be informix. On Windows, the owner must be a member of the Informix-Admin group.

---

The chunk pathname will not fit in the space specified.

**Cause:** The chunk pathname does not fit in the space that you specified.

**Action:** Choose a smaller size for the chunk, or free space where the chunk is to be created.

---

Cleaning stray LOs in sbspacenname.

**Cause:** The database server administrator is running onspaces -cl sbspacenname.

**Action:** None required.

---

Completed re-creating indexes.

**Cause:** The database server finished re-creating the deferred indexes.

**Action:** None required.

---

Configuration has been grown to handle up to integer chunks.

**Cause:** The database server administrator increased the number of chunks to the specified value by changing CONFIGSIZE or setting MAX_CHUNKS to a higher value.

**Action:** None required. The change was successful.

---

Configuration has been grown to handle up to integer dbslices.

**Cause:** The database server administrator increased the number of dbslices to the specified value by changing CONFIGSIZE or setting MAX_DBSLICES to a higher value.

**Action:** None required. The change was successful.

---

Configuration has been grown to handle up to integer dbspaces.

**Cause:** The database server administrator increased the number of dbspaces to the specified value by changing CONFIGSIZE or setting MAX_DBSPACES to a higher value.

**Action:** None required. The change was successful.

---

Continuing Long Transaction (for COMMIT): tx 0xn.

**Cause:** The logical log has filled beyond the long-transaction high-watermark (LTXHWM), but the offending long transaction is in the process of committing. In this case, the transaction is permitted to continue writing to the logical log and is not rolled back. The address of the transaction structure in shared memory is displayed as hexadecimal value tx 0xn.

**Action:** None required.
Could not disable priority aging: errno = number.

Cause: An operating-system call failed while it was trying to disable priority aging for the CPU virtual processor. The system error number associated with the failure is returned.

Action: See your operating-system documentation.

Could not fork a virtual processor: errno = number.

Cause: The fork of a virtual processor failed. The database server returns the operating-system error number associated with the failure.

Action: For information on determining the maximum number of processes available per user and for the system as a whole, refer to your operating-system documentation.

Create vp: cannot allocate memory.

Cause: The database server cannot allocate new shared memory.

Action: The database server administrator must make more shared memory available. This situation might require increasing SHMTOTAL or reconfiguring the operating system. This message is usually accompanied by other messages that give additional information.

Messages: D-E-F

Dataskip is OFF for all dbspaces.

Cause: Informational.

Action: None required.

Dataskip is ON for all dbspaces.

Cause: Informational.

Action: None required.

Dataskip is ON for dbspaces: dbspacelist.

Cause: Informational; DATASKIP is ON for the specified dbspaces.

Action: None required.

Dataskip will be turned {ON | OFF} for dbspace name.

Cause: Informational; DATASKIP is ON or OFF for the specified dbspace.

Action: None required.

DBSERVERALIASES exceeded the maximum limit of 32

Cause: The limit of 32 aliases was reached.

Action: Nothing. Only the first 32 will be used.

DBSPACETEMP internal list not initialized, using default.

Cause: An error occurred while initializing a user-specified DBSPACETEMP list. Typically this condition is due to a memory-allocation failure.

Action: Check for accompanying error messages.

The DBspace/BLOBspace spacename is now mirrored.

Cause: You successfully added mirroring to the indicated storage space.

Action: None required.

The DBspace/BLOBspace spacename is no longer mirrored.

Cause: You have ended mirroring for the indicated storage space.

Action: None required.

Dbname: write failed, file system is full.

Cause: Because the file system dbname is full, the write failed.

Action: Free some space in dbname.

Dropping temporary tblspace 0xn, recovering nn pages.

Cause: During shared-memory initialization, the database server routinely searches for temporary tables that are left without proper cleanup. If the database server finds a temporary table, it drops the table and recovers the space. The database server located the specified temporary tblspace and dropped it. The value 0xn is the hexadecimal representation of the tblspace number.

Action: None required.
Dynamically allocated new shared memory segment (size nnn).

Cause: This status message informs you that the database server successfully allocated a new shared-memory segment of size nnn.

Action: None required.

Error writing shmem to file filename error.
Unable to create output file filename errno=nn.
Error writing filename errno=nn.

Cause: The database server detected an error in an attempt to write shared memory to filename. The first message is followed by one of the next two. Either the attempt failed because the output file could not be created or because the contents of shared memory could not be written. The error refers to the operating-system error that prompted the attempted write of shared memory to a file. The value of nn is the operating-system error.

Action: See your operating-system documentation.

Fail to extend physical log space.

Cause: The attempt to extend the physical log space failed. Either the path does not exist or the permissions are incorrect.

Action: Use a path that exists. Check permissions on the current working directory. You or the system administrator must give your group execute permission on the current working directory. After your group has been given permission, retry the operation that generated this message.

Fatal error initializing CWD string.
Check permissions on current working directory.
Group groupname must have at least execute permission on ‘.’.

Cause: Group groupname does not have execute permission for the current working directory.

Action: Check permissions on the current working directory. You or the system administrator must give your group execute permission on the current working directory. After your group has been given permission, retry the operation that generated this message.

The following tables have outstanding old version data
pages due to an In-Place Alter Table. Perform
UPDATE tablename SET column = column WHERE 1=1;
to clear these pages from the following tables.

Cause: Reversion to a previous version of the database server has been attempted while an in-place ALTER TABLE is in progress. The previous versions of the database server cannot handle tables that have multiple schemas of rows in them.

Action: Force any in-place alters to complete by updating the rows in the affected tables before you attempt to revert to a previous version of the database server. To do this, create a dummy update in which a column in the table is set to its own value, forcing the row to be updated to the latest schema in the process without actually changing column values. Rows are always altered to the latest schema, so a single pass...
through the table that updates all rows completes all outstanding in-place alters.

**Fragments** `dbname1 dbname2` **of** `table` `tablename` **set to non-resident.**

**Cause:** The specified fragments of `tablename` either have been set to nonresident by the SET TABLE statement.

**Action:** None required.

**Forced-resident shared memory not available.**

**Messages: G-H-I**

**gcore** `pid`; `mv` `core.pid` `dir/core.pid.ABORT`.

**Cause:** This status message during a database server failure provides the name and place of each core file associated with the virtual processors.

**Action:** None required.

**I/O function** `chunk mn, pagenum mn, pagecnt na --> errno = bb`.

**Cause:** An operating-system error occurred during an attempt to access data from disk space. The operating-system function that failed is defined by `function`. The chunk number and physical address of the page where the error occurred are displayed as integers. The `pagcnt` value refers to the number of pages that the thread was attempting to read or write. If an `errno` value is displayed, it is the number of the operating-system error and might explain the failure. If `function` is specified as `bad request`, some unexpected event caused the I/O attempt on an invalid chunk or page.

**Action:** If the chunk status changes to 0, or down, restore the chunk from its mirror or repair the chunk. Otherwise, perform a data restore.

**I/O error, primary/mirror Chunk pathname -- Offline (sanity).**

**Cause:** The database server detected an I/O error on a primary or mirror chunk with `pathname`. The chunk was taken offline.

**Action:** Check that the device on which the chunk was stored is functioning as intended.

Deleted indexes idx1 and idx 2 error message

**Informix database_server Initialized - Complete Disk Initialized.**

**Cause:** Disk space and shared memory have been initialized. Any databases that existed on the disk before the initialization are now inaccessible.

**Action:** None required.

**Freed mm shared-memory segment(s) number bytes.**

**Cause:** The database server sends this message to the message log after you run the `-F` option of the `onmode` utility to free unused memory. The message informs you of the number of segments and bytes that the database server successfully freed.

**Action:** None required.

**Informix database_server Initialized - Shared Memory Initialized.**

**Cause:** Shared memory has been initialized.

**Action:** None required.

**Informix database_server Stopped.**

**Cause:** The database server has moved from quiescent mode to offline mode. The database server is offline.

**Action:** None required.

**ERROR: Insufficient available disk in the root dbspace to increase the entire Configuration save area.**

**Cause:** The user attempted to increase the number of storage objects to a specific value by changing `CONFIGSIZE` or setting MAX_DBSPACES, MAX_DSBLICES, or MAX_CHUNKS to a higher value, but the database server did not have enough rootspace for the increased number of storage objects. A storage object might be a dbspace, dsblice, or chunk.

**Action:** Increase the size of the root dbspace or reset `CONFIGSIZE`,`MAX_DBSPACES`, `MAX_DSBLICES`, or `MAX_CHUNKS` to a lower value and restart the database server. For example, if you set `MAX_CHUNKS` to 32,768, but the root dbspace did not have enough space, set `MAX_CHUNKS` to a lower value.

Insufficient available disk in the root dbspace for the CM save area. Increase the size of the root dbspace in the ONCONFIG file and reinitialize the server.

**Cause:** The cause might be one of the following:

- The user attempted to increase the number of storage objects to a specific value by changing `CONFIGSIZE` or setting MAX_DBSPACES, MAX_DSBLICES, or MAX_CHUNKS to a higher value, but the database server did not have enough rootspace for the
increased number of storage objects. A storage object might be a dbspace, dbslice, or chunk.

• The user converted to a database server version that requires slightly more rootspace, but it is not available (this case is unlikely).

**Action:** Take one of the following actions:

• Increase the size of the root dbspace or reset CONFIGSIZE, MAX_DBSPACES, MAX_DBSLICES, or MAX_DBSLICES to a lower value and restart the database server. For example, if you set MAX_DBSPACES to 32,768 but the root dbspace did not have enough space, set MAX_DBSPACES to a lower value.

• Increase the size of the root dbspace and reinitialize the database server.

**Internal overflow of shmids, increase system max shared memory segment size.**

**Cause:** The database server was initializing shared memory when it ran out of internal storage for the shared-memory IDs associated with this segment.

**Action:** Increase the value of your maximum kernel shared-memory segment size, usually SHMMAX. For more information, see your operating-system documentation.

**Messages: J-K-L-M**

---

**Listener-thread err = error_number; error_message.**

**Cause:** A listener thread has encountered an error. This message displays the error number and message text.

**Action:** For the cause and corrective action, see the IBM Informix Online Documentation site at [http://www.ibm.com/software/data/informix/pubs/library/](http://www.ibm.com/software/data/informix/pubs/library/)

---

**Lock table overflow - user id mm session id nn.**

**Cause:** A thread attempted to acquire a lock when no locks were available. The user ID and session ID are displayed.

**Action:** Increase the LOCKS configuration parameter, and initialize shared memory.

---

**Logical-log File not found.**

**Cause:** The checkpoint record in the root dbspace reserved page is corrupted.

**Action:** Perform a data restore from dbspace backup.

---

**Logical Log mm Complete.**

**Cause:** The logical-log file identified by log-ID number mm is full. The database server automatically switches to the next logical-log file in the sequence.

**Action:** None required.

---

**Logical logging verror for typesubtype in (failed_system).**

**Cause:** Logging failed. The log record that caused the error is identified as follows:

- **type** Is the logical-log record type.
- **subtype** Is the logging subsystem.
- **nn** Is the logical-log ID where the record is stored.
- **0xn** Is the hexadecimal address position within the log.

---

**Log record (typesubtype) at log mm, 0xn was not undone.**

**Cause:** A log undo failed because a log is corrupt.

The log record that caused the error is identified as follows:

- **type** Is the logical-log record type.
- **subtype** Is the logging subsystem.
- **nn** Is the logical-log ID where the record is stored.
- **0xn** Is the hexadecimal address position within the log.
Action: To determine if any action is needed, use the onlog utility to examine the logical log. Contact Technical Support.

| Log record (type:subtype) failed, partnum pnum row rid iserno num. |
| Cause: A logging failure occurred. The log record that caused the error is identified as follows: type Is the logical-log record type. subtype Is the logging subsystem. pnum Is the part number. rid Is the row ID. num Is the iserror number. Action: Contact Technical Support. |

| Log record (type:subtype) in log mn, offset 0xn was not rolled back. |
| Cause: A log undo failed because a log is corrupt. The log record that caused the error is identified as follows: type Is the logical-log record type. subtype Is the logging subsystem. log Is the logical-log ID where the record is stored. offset Is the hexadecimal address position within the log. Action: To determine if any action is needed, use the onlog utility to examine the logical log. Contact Technical Support. |

| Logical Recovery allocating nn worker threads thread_type. |
| Cause: The database server determined the number of worker threads that will be used for parallel recovery. The variable thread_type can assume the values ON_RECVRY_THREADS or OFF_RECVRY_THREADS. Action: This status message requires no action. If you want a different number of worker threads allocated for parallel recovery, change the value of the ONCONFIG configuration parameter ON_RECVRY_THREADS or OFF_RECVRY_THREADS. |

| Logical Recovery Started. |
| Cause: Logical recovery began. Action: This status message requires no action. |

| Maximum server connections number. |
| Cause: Outputs with each checkpoint message to indicate the maximum number of concurrent connections to the database server since the last restart. Action: This message helps the customer track license usage to determine when more licenses need to be purchased. For assistance, Contact Technical Support. |

| Memory allocation error. |
| Cause: The database server ran out of shared memory. Action: Take one of the following actions: Increase swap space on the computer. Check kernel shared-memory parameters for limits on shared memory. Decrease the size of the memory allocated, with the buffers field in the BUFFERPOOL configuration parameter. Increase the virtual-memory size (SHMVIRTSIZE), the size of the added segments, (SHMADD), or your total shared-memory size (SHMTOTAL). |

| Mirror Chunk chunkname added to space spacename. Perform manual recovery. |
| Cause: Fast recovery, full recovery, or an HDR secondary has recovered the add of a mirror chunk. It does not perform automatic mirror recovery, however. The administrator must do this. Action: Use either the onspaces utility or ON–Monitor to attempt to recover the mirror chunks. |

| Mixed transaction result. (pid=nn user=userid) |
| Cause: You receive this message only when more than one database server is involved in a transaction. This message indicates that a database server, after preparing a transaction for commit, heuristically rolled back the transaction, and the global transaction completed inconsistently. The pid value is the user-process identification number of the coordinator process. The value of user is the user ID associated with the coordinator process. Action: See the information on recovering manually from failed two-phase commit in your IBM Informix Administrator’s Guide. |

| mt_shm_free_pool: pool 0xn has blocks still used (id nn). |
| Cause: An internal error occurred during a pool deallocation because blocks are still associated with the pool. Action: Contact Technical Support. |
**mt_shm_init: can’t create resident/virtual segment.**

**Cause:** The causes for the failure to create the resident or virtual segment are as follows: (1) the segment size is less than the minimum segment size; (2) the segment size is larger than the maximum segment size; (3) allocating another segment would exceed the allowable total shared-memory size; or (4) a failure occurred while the database server was trying to allocate the segment.

**Action:** If you suspect that this error was generated because of item 1 or 2 in the preceding paragraph, Contact Technical Support. To correct item 3, increase the SHMTOTAL value in your ONCONFIG configuration file. For additional information about errors generated because of item 4, see your logical-log file.

**mt_shm_remove: WARNING: may not have removed all/correct segments.**

**Cause:** When the operating system tried to remove the shared-memory segments associated with the database server, the last segment did not equal the last segment registered internally. This situation is probably due to the unexpected failure of the database server.

**Action:** Remove any segments that were not cleaned up.

### Messages: N-O-P

**Newly specified value of value for the pagesize in the configuration file does not match older value of value. Using the older value.**

**Cause:** This message displays upon database server restart. The PAGESIZE value changed in the ONCONFIG file after the database server was initialized.

**Action:** The database server uses the older PAGESIZE value.

**Not enough main memory.**

**Cause:** The database server detected an error in an attempt to acquire more memory space from the operating system.

**Action:** For more information about shared-memory configuration and management, refer to your operating-system documentation.

**Not enough logical-log files, Increase LOGFILES.**

**Cause:** During a data restore, the value of the LOGFILES configuration must always be greater than or equal to the total number of logical-log files. At some point during the restore, the number of logical-log files exceeded the value of LOGFILES.

**Action:** Increase the value of LOGFILES in ONCONFIG.

**Not enough physical procs for affinity.**

**Cause:** The ONCONFIG parameters AFF_NPROCS and AFF_SPROC are not correctly set. AFF_SPROC plus AFF_NPROCS is greater than the number of physical processors on your computer or node.

**Action:** Reset AFF_NPROCS and AFF_SPROC, such that the value AFF_SPROC plus value of AFF_NPROCS is less than or equal to the number of physical processors.

**The number of configured CPU poll threads exceeds NUMCPUVPS.**

**Cause:** The number of in-line poll threads that you specified in the ONCONFIG configuration file exceeds the number of CPU virtual processors.

**Action:** Reduce the number of in-line poll threads to be less than or equal to the number of CPU virtual processors.

**onconfig parameter parameter modified from old_value to new_value.**

**Cause:** When the database server shared memory is reinitialized, this message documents any changes that occurred since the last initialization.

**Action:** None required.

**oninit: Cannot have SINGLE_CPU_VP non-zero and number of CPU VPs greater than 1.**

**Cause:** The ONCONFIG file contains VPCLASS cpu with a num= value greater than 1 and a nonzero value for SINGLE_CPU_VP. SINGLE_CPU_VP must be 0 (or omitted) when there are more than 1 CPU VPs.

**Action:** Correct the ONCONFIG file and restart the database server.

**oninit: Cannot have SINGLE_CPU_VP non-zero and user-defined VP classes.**

**Cause:** The ONCONFIG file contains a user-defined VPCLASS as well as a nonzero value for SINGLE_CPU_VP. SINGLE_CPU_VP must be 0 (or omitted) when the ONCONFIG file contains a user-defined VPCLASS.

**Action:** Correct the ONCONFIG file and restart the database server.

**oninit: Cannot mix VPCLASS cpu and NUMCPUVPS,**
AFF_SPROC, AFF_NPROCS, or NOAGE parameters.

**Cause:** The ONCONFIG file contains both VPCLASS cpu and one or more of the other listed parameters. It cannot contain both.

**Action:** Correct the ONCONFIG file and restart the database server.

---

**oninit:** Cannot mix VPCLASS aio and NUMAIOVPS parameters.

**Cause:** The ONCONFIG file contains both VPCLASS aio and NUMAIOVPS. It cannot contain both.

**Action:** Correct the ONCONFIG file and restart the database server.

---

**oninit:** Fatal error in initializing ASF with ‘ASF_INIT_DATA’ flags ascname = ‘25507’.

**Cause:** The nettype value specified in the sqlhosts file or registry for the database server is invalid or unsupported, or the servicename specified in the sqlhosts file or registry for the database server is invalid.

**Action:** Check the nettype and servicename values in the sqlhosts file or registry for each DBSERVERNAME and for the DBSERVERALIASES. Check the nettype value in each NETTYPE parameter in the ONCONFIG file.

---

**oninit:** invalid or missing name for Subsystem Staging Blobspace.

**Cause:** You set the configuration parameter STAGEBLOB to a blobspace that does not exist.

**Action:** Use the -d option of onspaces to create the blobspace specified in STAGEBLOB, and restart the database server.

---

**oninit:** Too many VPCLASS parameters specified.

**Cause:** Too many VPCLASS parameter lines have been specified in the ONCONFIG file.

**Action:** Reduce the number of VPCLASS lines, if possible. If not possible, contact Technical Support.

---

**oninit:** VPCLASS classname duplicate class name.

**Cause:** The VPCLASS classname in the ONCONFIG file has a duplicate name. VP class names must be unique.

**Action:** Correct the duplicate name and restart the database server.

---

**oninit:** VPCLASS classname illegal option.

**Cause:** One of the fields in the VPCLASS classname parameter is illegal.

**Action:** Correct the parameter in the ONCONFIG file and restart the database server.

---

**oninit:** VPCLASS classname maximum number of VPs is out of the range 0-10000.

**Cause:** The maximum number of VPs specified by a VPCLASS parameter line must be in the range 1 to 10,000.

**Action:** Correct the value and restart the database server.

---

**oninit:** VPCLASS classname name is too long.

**Maximum length is maxlength.**

**Cause:** The length of the name field in VPCLASS classname is too long.

**Action:** Choose a shorter class name, correct the ONCONFIG file, and restart the database server.

---

**oninit:** VPCLASS classname number of VPs is greater than the maximum specified.

**Cause:** The initial number of VPs specified by a VPCLASS parameter is greater than the maximum specified by the same VPCLASS parameter.

**Action:** Correct the VPCLASS parameter and restart the database server.

---

**oninit:** VPCLASS classname number of VPs is out of the range 0-10000.

**Cause:** The initial number of VPs specified by a VPCLASS parameter line must be in the range 1 to 10,000.

**Action:** Reduce the number of VPCLASS lines, if possible. If not possible, contact Technical Support.

---

**onmode:** VPCLASS classname name is too long.

**Maximum length is maxlength.**

**Cause:** The name of a dynamically added VP class that onmode -p specifies is too long.

**Action:** Choose a shorter name, and retry the onmode -p command.
Optical Subsystem is running.

**Cause:** You set the value of the STAGEBLOB parameter in the configuration file, and the database server is communicating properly with the optical-storage subsystem.

**Action:** No action is required.

Optical Subsystem is not running.

**Cause:** You set the value of the STAGEBLOB parameter in the configuration file, but the database server cannot detect the existence of the optical-storage subsystem.

**Action:** Check that the optical subsystem is online.

Optical Subsystem STARTUP Error.

**Cause:** The database server detects that the optical-storage subsystem is running, but the database server cannot communicate with it properly.

**Action:** Check your optical subsystem for errors.

Online Mode.

**Cause:** The database server is in online mode. Users can access all databases.

**Action:** This status message requires no action.

**onspaces: unable to reset dataskip.**

**Cause:** This error message comes from the onspaces utility. For some reason, the utility cannot change the specification of DATASKIP (ON or OFF) across all dbspaces in the database server instance.

**Action:** You are unlikely to receive this message. If the error persists after you restart the database server, Contact Technical Support.

Open transaction detected when changing log versions.

**Cause:** The database server detected an open transaction while it was trying to convert the data from a previous version of the database server.

**Action:** Conversion is not allowed unless the last record in the log is a checkpoint. You must restore the previous version of the database server, force a checkpoint, and then retry conversion.

Out of message shared memory.

**Cause:** The database server could not allocate more memory for the specified segment.

**Action:** For additional information, see the log file.

Out of resident shared memory.

**Cause:** The database server could not allocate more memory for the specified segment.

**Action:** For additional information, see the log file.

Out of virtual shared memory.

**Cause:** The database server could not allocate more memory for the specified segment.

**Action:** For additional information, see the log file.

**PANIC: Attempting to bring system down.**

**Cause:** A fatal database server error occurred.

**Action:** See the error that caused the panic and attempt the corrective action suggested by the error message. For additional information that might explain the failure, refer also to other messages in the message-log file.

**Participant site database_server heuristically rolled back.**

**Cause:** A remote site rolled back a transaction after it reached the prepared-for-commit phase.

**Action:** You might need to roll back the transaction on other sites and then restart it.

**Physical recovery complete: number pages examined, number pages restored.**

**Cause:** This message displays during fast recovery.

The **number of pages examined** indicates the number of page images that exist in the physical log. The **number of pages restored** indicates the actual number of pages that are restored from the physical log. The number of pages restored is always less than or equal to the number examined.

The database server might physically log a page image multiple times between checkpoints. Physical recovery restores only the first logged page image.

If a page stays in the memory buffer pool, the database server physically logs it once per checkpoint, and stores one page image in the physical log. If the buffer pool is too small, a page that is being updated many times might get forced out of the buffer pool to disk and then brought back into memory for the next update. Each time the page is brought into memory, it is physically logged again, resulting in duplicate page images in the physical log.

**Action:** If the **number of pages examined** is much larger than the **number of pages restored**, increase the size of the buffer pool to reduce the number of duplicate before-images. For more information, see the IBM Informix Performance Guide.
Physical recovery started at page (chunk offset).

Cause: This message displays during fast recovery. Chunk is the number of the chunk that contains the physical log. Offset is the page offset of the start of the physical log entries. Physical recovery begins restoring pages from that point.

Action: No action required. For information on fast recovery, see the IBM Informix Administrator’s Guide.

Portions of partition partnum of table tablename in database dbname were not logged. This partition cannot be rolled forward.

Cause: Light appends occurred to the operational table since the last backup.

Action: If you want full access to data in this table, you need to alter the table to raw and then to the desired table type. This alter operation removes inconsistencies in the table that resulted from replaying non-logged operations such as light appends.

Possible mixed transaction result.

Cause: This message indicates that error -716 has been returned. Associated with this message is a list of the database servers where the result of a transaction is unknown.

Action: For information on determining if a transaction was implemented inconsistently, see the IBM Informix Administrator’s Guide.

Prepared participant site server_name did not respond.

Cause: Too many attempts were made to contact remote site server_name. After several timeout intervals were met, the site was determined to be down.

Action: Verify that the remote site is online and that it is correctly configured for distributed transactions. Once the remote site is ready, reinitiate the transaction.

Prepared participant site server_name not responding.

Cause: The database server is attempting to contact remote site server_name. For some unknown reason, the database server cannot contact the remote site.

Action: Verify that the remote site is online and that it is correctly configured for distributed transactions.

Messages: Q-R-S

Quiescent Mode.

Cause: The database server has entered quiescent mode from some other state. On UNIX, only users logged in as informix or as root can interact with the database server. On Windows, only members of the Informix-Admin group can interact with the database server. No user can access a database.

Action: None required.

Read failed. Table name, Database name, iserro = number

Cause: An error occurred reading the specified system table.

Action: Note the error number and contact Technical Support.

Recovery Mode.

Cause: The database server entered the recovery mode. No user can access a database until recovery is complete.

Action: None required.

Recreating index: ‘dbname’owner’.tablename-idxname’.

Cause: This message indicates which index is currently being re-created.

Action: None required.

Rollforward of log record failed, iserro = nn.

Cause: The message appears if, during fast recovery or a data restore, the database server cannot roll forward a specific logical-log record. The database server might be able to change to quiescent or online mode, but some inconsistency could result. For further information, see the message that immediately precedes this one. The iserro value is the error number.

Action: Contact IBM Informix Technical Support.

Root chunk is full and no additional pages could be allocated to chunk descriptor page.

Cause: The root chunk is full.

Action: To free space in the root chunk, take one of the following actions:

• Drop and re-create the sysmaster database.
• Move user tables from the root dbspace to another dbspace.
• Refragment tables.

scan_logundo: subsys ss, type tt, iserro ee.

Cause: A log undo failed because log type tt is corrupt.
The variables in this message have the following values:

- ss: Is the subsystem name.
- tt: Is the logical-log record type.
- ee: Is the iserror number.

**Action:** Examine the logical log with the onlog utility to determine if any action is needed. Contact Technical Support.

---

**Session completed abnormally. Committing tx id 0xn, flags 0xn.**

**Cause:** Abnormal session completion occurs only when the database server is attempting to commit a transaction that has no current owner, and the transaction develops into a long transaction. The database server forked a thread to complete the commit.

**Action:** None required.

---

**Session completed abnormally. Rolling back tx id 0xn, flags 0xn.**

**Cause:** Abnormal session completion occurs only when the database server is attempting to commit a transaction that has no current owner, and the transaction develops into a long transaction. The database server forked a thread that rolled back the transaction.

**Action:** None required.

---

**shmctl: errno = nn.**

**Cause:** An error occurred while the database server tried to remove or lock a shared-memory segment. The operating-system error number is returned.

**Action:** See your operating-system documentation.

**shmctl: errno = nn.**

**Cause:** An error occurred while the database server was trying to detach from a shared-memory segment. The operating-system error number is returned.

**Action:** See your operating-system documentation.

---

**shmem sent to filename.**

**Cause:** The database server wrote a copy of shared memory to the specified file as a consequence of an assertion failure.

**Action:** None.

---

**shmem get: some_str os_errno: key shmkey: some_string.**

**Cause:** Either the creation of a shared-memory segment failed, or an attempt to get the shared-memory ID associated with a certain key failed. The system error number and the suggested corrective action are returned.

**Action:** Consult your operating-system documentation.

---

**Shutdow (onmode -k) or override (onmode -O).**

**Cause:** A dbspace has gone down during a checkpoint interval. The database server is configured to wait for an override when this situation occurs.

**Action:** Either shut down the database server or issue an onmode -O command to override the down dbspace. For more information on the onmode utility, see “In This Chapter” on page 11-1.

---

**Shutdow Mode.**

**Cause:** The database server is in the process of moving from online mode to quiescent mode.

**Action:** None required.

---

**Space spacename added.**

**Cause:** The database server administrator added a new storage space spacename to the database server.

**Action:** None required.
**Space spacename dropped.**

**Cause:** The database server administrator dropped a storage space `spacename` from the database server.

**Action:** None required.

**Space spacename -- Recovery Begins(addr).**

**Cause:** This informational message indicates that the database server is attempting to recover the storage space.

The variables in this message have the following values:

- `spacename` Is the name of the storage space that the database server is recovering.
- `addr` Is the address of the control block.

**Action:** None required.

**Space spacename -- Recovery Complete(addr).**

**Cause:** This informational message indicates that the database server recovered the storage space.

The variables in this message have the following values:

- `spacename` Is the name of the storage space that the database server has recovered.
- `addr` Is the address of the control block.

**Action:** None required.

**Space spacename -- Recovery Failed(addr).**

**Cause:** This informational message indicates that the database server was unable to recover the storage space.

The variables in this message have the following values:

- `spacename` Is the name of the storage space that the database server failed to recover.
- `addr` Is the address of the control block.

**Action:** None required.

---

**Successfully extend physical log space**

**Cause:** The physical log space was successfully extended to the file `plog_extend.servernum` under the designated path.

**Action:** None required.

---

**sysmaster database built successfully.**

**Cause:** The database server successfully built the sysmaster database.

**Action:** None required.

---

**Messages: T-U-V**
This ddl operation is not allowed due to deferred constraints pending on this table and dependent tables.

**Cause:** This error is returned when you attempt to start a violations table and constraints are in deferred mode.

**Note:** No error is returned if you start a violations table and then later set the constraints to deferred. However, the violations get undone immediately rather than written into the deferred constraint buffer. For more information, see the IBM Informix Guide to SQL: Syntax.

**Action:** If you would like to start a violations table, you must either change the constraint mode to immediate or commit the transaction.

This type of space does not accept log files.

**Cause:** Adding a logical-log file to a blobspace or sbspace is not allowed.

**Action:** Add the logical-log file to a dbspace. For more information, see the `onparams -a -d dbspace: Add a logical-log file` on page 13-2.

**TIMER VP: Could not redirect I/O in initialization, errno = nn.**

**Cause:** The operating system could not open the null device or duplicate the file descriptor associated with the opening of that device. The system error number is returned.

**Action:** See your operating-system documentation.

**Too Many Active Transactions.**

**Cause:** During a data restore, there were too many active transactions. At some point during the restore, the number of active transactions exceeded 32 kilobytes.

**Action:** None.

**Too many violations.**

**Cause:** The number of violations in the diagnostics table exceeds the limit that is specified in the MAX VIOLATIONS clause of the START VIOLATIONS TABLE statement. When a single statement on the target table (such as an INSERT or UPDATE statement) inserts more records into the violations table than the limit that is specified by the MAX VIOLATIONS clause, this error is returned to the user who issued the statement on the target table.

This MAX VIOLATIONS limit applies to each coserver. For example, if you reach the MAX VIOLATIONS limit on coserver z, you can continue to issue statements that violate rows on other coservers until you reach the MAX VIOLATIONS limit.

**Action:** To resolve this error, perform one of the following actions:

- Omit the MAX VIOLATIONS clause in the START VIOLATIONS TABLE statement when you start a violations table. Here, you are specifying no limit to the number of rows in the violations table.
- Set MAX VIOLATIONS to a high value.

**Transaction Not Found.**

**Cause:** The logical log is corrupt. This situation can occur when a new transaction is started, but the first logical-log record for the transaction is not a BEGWORK record.

**Action:** Contact Technical Support.

**Transaction heuristically rolled back.**

**Cause:** A heuristic decision occurred to roll back a transaction after it completed the first phase of a two-phase commit.

**Action:** None required.

**Transaction table overflow - user id nn, process id nn.**

**Cause:** A thread attempted to allocate an entry in the transaction table when no entries in the shared-memory table were available. The user ID and process ID of the requesting thread are displayed.

**Action:** Try again later.

**Unable to create output file filename errno = nn.**

**Cause:** The operating system cannot create output file filename. The errno is the number of the operating-system error returned.

**Action:** Verify that the directory exists and has write permissions.

**Unable to extend nn reserved pages for purpose in root chunk.**

**Cause:** The operating system cannot extend to nn reserved pages for purpose in root chunk. (The value purpose can be either Checkpoint/Log, DBSpace, Chunk, or Mirror Chunk.)

**Action:** Reduce the ONCONFIG parameter for the resource cited; bring the database server up and free some space in the primary root chunk. Then reattempt the same operation.

**Unable to initiate communications with the Optical Subsystem.**

**Cause:** The optical driver supplied by the optical-drive vendor has indicated that the drive is not accessible.

**Action:** Check driver installation and cabling between
Unable to start SQL engine.

**Cause:** The database server encountered an out-of-memory condition.

**Action:** No action is necessary.

Unable to open tblspace \text{nn}, iserrno = \text{nn}.

**Cause:** The database server cannot open the specified tblspace. (The value \text{nn} is the hexadecimal representation of the tblspace number.)

**Action:** See the ISAM error message number \text{nn}, which should explain why the tblspace cannot be accessed. The error message appears in *IBM Informix Error Messages* at the IBM Informix Online Documentation site at: www.ibm.com/software/data/developer/informix.

The value of pagesize \text{pagesize} specified in the config file is not a valid pagesize. Use 2048, 4096 or 8192 as the value for PAGESIZE in the onconfig file and restart the server.

**Cause:** This message displays upon disk initialization. The value of PAGESIZE that was specified in the ONCONFIG file is not a valid value.

**Action:** Restart the database server with a valid PAGESIZE value.

Violations table is not started for the target table.

**Cause:** If you issue a STOP VIOLATIONS TABLE statement for which no violations table is started, you receive this message.

**Action:** To recover from this error, you must start a violations table for the target table.

Violations table reversion test completed successfully.

**Cause:** This message is recorded in the logmessage table in the systmaster database when the revtestviolations.sh script has completed successfully (no open violations tables were found).

**Action:** No action is necessary. For more information on revtestviolations.sh, see the *IBM Informix Migration Guide*.

Violations table reversion test failed.

**Cause:** When the database server finds an open violations table, it reports errors 16992 and 16993 in the logmessage table in the systmaster database and aborts the reversion process.

**Action:** When this message appears, you must issue the STOP VIOLATIONS TABLE FOR \text{table_name} command for each open violations table. After you close all open violations tables, you can restart the reversion process.

Violations table reversion test start.

**Cause:** This message is recorded in the logmessage table in the systmaster database when the revtestviolations.sh script is executed.

**Action:** No action is necessary. For more information on revtestviolations.sh, see the *IBM Informix Migration Guide*.

Violations tables still exist.

**Cause:** This message is recorded in the logmessage table in the systmaster database when an open violations table is found.

**Action:** When this message appears, you must issue the STOP VIOLATIONS TABLE FOR \text{table_name} command for each open violations table. After you close all open violations tables, you can restart the reversion process.

Virtual processor limit exceeded.

**Cause:** You configured the database server with more than the maximum number of virtual processors allowed (1000).

**Action:** To reduce the number of virtual processors, decrease the values of VPCLASS, NUMCPUVPS, NUMAIOVPS, or NETTYPE in your ONCONFIG file.

**Cause:** This message indicates an internal error.

**Action:** Contact Technical Support.

**Action:** Contact Technical Support.

VPCLASS class name is too long. Maximum length is maxlength.

**Cause:** This message indicates an internal error.

**Action:** Contact Technical Support.

VPCLASS class name duplicate class name.

**Cause:** The physical processors in the affinity specification for the VP class class name do not exist or are offline. The problem might be with the VPCLASS parameter for cpu class VPs or with the AFF_SPROC and AFF_NPROCS parameters.

**Action:** Make sure the named processors are online. Correct the affinity specification for the named VP class. Restart the database server.
Messages: W-X-Y-Z

WARNING: aio_wait: errno = nn.
Cause: While the database server was waiting for an I/O request to complete, it generated error number nn on an operation that it was attempting to execute.
Action: Contact Technical Support for assistance.

WARNING: Buffer pool size may cause database server to get into a locked state. Recommended minimum buffer pool size is num times maximum concurrent user threads.
Cause: There are not enough buffers in the buffer pool. The database server could use all available buffers and cause a deadlock to occur.
Action: Change the buffers field in the BUFFERPOOL parameter in the ONCONFIG file to the number that this message recommends. For more information on the BUFFERPOOL parameter, see "BUFFERPOOL" on page 1-18.

warning: Chunk time stamps are invalid.
Cause: A sanity check is performed on chunks when they are first opened at system initialization. The chunk specified did not pass the check and will be brought offline.
Action: Restore the chunk from a dbspace backup or its mirror.

Warning: name_old is a deprecated onconfig parameter. Use name_new instead. See the release notes and the Informix Administrator's Reference for more information.
Cause: A deprecated ONCONFIG parameter was used. This message displays the first time that you use a deprecated parameter. The shorter form of the message displays thereafter.

Action: Use the suggested alternative ONCONFIG parameter.

Warning: name_old is a deprecated onconfig parameter. Use name_new instead.
Cause: A deprecated ONCONFIG parameter was used.
Action: Use the suggested alternative ONCONFIG parameter.

WARNING: Unable to allocate requested big buffer of size nn.
Cause: The internal memory allocation for a big buffer failed.
Action: Increase either virtual memory size (SHMVIRTSIZE), the size of the added segments (SHMADD), or your total shared-memory size (SHMTOTAL).

You are turning off smart large object logging.
Cause: These changes will become the new sbspace default values. Changes have been made to the sbspace. The onspaces utility will read and update 100 smart large objects at a time and commit each block of 100 smart large objects as a single transaction. This utility might take a long time to complete.
Action: This informational message occurs when you issue the following command:

```sh
onspaces -ch sbspace -Df "LOGGING=OFF" -y
```

For more information, see "onspaces -ch: Change sbspace default specifications" on page 14-17

Messages: Symbols

HH:MM:SS  Informix database server Version R.VV.PPPP  Software Serial Number RDS#XYYYYY.

Cause: This message indicates the start-up of the database server, after the initialization of shared memory.
Action: No action is required.

argument: invalid argument.
Cause: This internal error indicates that an invalid argument was passed to an internal routine.
Action: Contact Technical Support.

function_name: cannot allocate memory.
Cause: The database server cannot allocate memory from internal shared-memory pool.
Action: Increase either virtual-memory size (SHMVIRTSIZE), the size of the added segments (SHMADD), or your total shared-memory size (SHMTOTAL).
Conversion/Reversion Messages

These messages might display during database server conversion or reversion.

Messages: A-C

Cannot revert constraint with id id (in syschecks).
Cause: The database has a constraint that was defined in a version more recent than the one to which you are reverting.
Action: Drop the specified constraint and retry reversion.

Cannot revert new fragment expression for index index, tabid id.
Cause: The index fragmentation was defined in a version more recent than the one to which you are reverting.
Action: Drop the problem index-fragmentation scheme and retry reversion.

Cannot revert new table fragment expression for table with id id.
Cause: The fragmentation of this table was defined in a version more recent than the one to which you are reverting.
Action: Drop the problem table fragmentation scheme and retry reversion.

Cannot update page zero.
Cause: Attempt to write page zero failed.
Action: Contact Technical Support.

Checking database name for revertibility.
Cause: Indicates that start of the reversion checks on the specified database.
Action: None required.

Conversion of pre 7.3 in-place alter started status.
Cause: The database server is converting data structures for in-place alters to the new format.
Action: None required.

Conversion of pre 9.2 database tablespaces status.
Cause: The database server is converting tablespaces to the new format.
Action: None required.

The conversion of the database name has failed.
Cause: Indicates that the conversion of the specified database has failed.
Action: Connect to the database. This action triggers conversion of the database. If it fails, the relevant error message appears. Contact Technical Support.

Converting database name...
Cause: This message appears at the start of conversion of each database in the system.
Action: None required.

Converting in-place alters to new format.
Cause: The database server is converting data structures for in-place alters to the new format.
Action: None required.

Converting 'onpload' database...
Cause: Printed in online.log at the beginning of onpload conversion.
Action: None required.

Converting partition header from version 7.x.
Cause: The database server is converting the partition header page to the new format that contains the chunk number and offset.
This message is optional verbose output that is logged only if you start oninit with the -v flag.
Action: None required.

Converting partition header page address.
Cause: The database server is converting the partition header page to the new format that contains the chunk number and page offset.
This message is optional verbose output that is logged only if you start oninit with the -v flag.
Action: None required.

Converting partition header pages status.
Cause: This message tracks the progress of the conversion of the partition header pages. The status is identified as follows:
  • started
  • succeeded
• **FAILED**

**Action:** If the status is started or succeeded, no action is required.

If conversion of the partition header pages failed, restart the database server. It will attempt to continue converting where it left off in the restartable conversion phase. If this action fails, diagnose the problem, restore from tape, fix the problem, and retry conversion.

---

**Converting partition keys to 9.2.**

**Cause:** The database server is converting the partition keys to the Version 9.2 format.

---

**Messages: D-F**

The database name has been converted successfully.

**Cause:** Indicates successful completion of the conversion of the specified database.

**Action:** None required.

---

Database name is not revertible...

**Cause:** The database has failed one of the reversion checks and is not revertible.

**Action:** Take action to correct the error displayed as a separate message.

---

Database name is revertible...

**Cause:** The database has passed all reversion checks and is revertible to the specified version.

**Action:** None required.

---

Database name: Must drop trigger (id = id_number).

**Cause:** The database contains a trigger that was created in a version more recent than the one to which you are converting.

**Action:** Drop the trigger with the specified trigger identification number and then attempt reversion.

---

Database name SUCCESSFULLY reverted...

**Cause:** Indicates the success of reversion of the specified database.

**Action:** None required.

---

... dropping sysmaster database.

**Cause:** The database server is dropping sysmaster database during the reversion process.

**Action:** No action is required.

---

This message is optional verbose output that is logged only if you start oninit with the -v flag.

**Action:** None required.

---

Converting partition name for databaseset tablename.

**Cause:** The database server is converting the partition name for the databaseset tablename.

This message is optional verbose output that is logged only if you start oninit with the -v flag.

**Action:** None required.

---

The dummy updates failed while converting database name. This may imply data corruption in the database. If so, restore the original database with the tape backup. For more information, see output_file.

**Cause:** During conversion of a database from a version earlier than Version 9.2, dummy update statements are run against the system tables in the database being converted. This message indicates failure in running one of these update statements.

**Action:** To retry the dummy updates, run the dummy update script for your old database server version. For instructions, refer to the IBM Informix Migration Guide.

If data corruption occurred, restore the original database with the tape backup. For more information, see the IBM Informix Backup and Restore Guide.

---

The dummy updates succeeded while converting database name.

**Cause:** During conversion of a database from a version earlier than Version 9.2, dummy update statements are run against the system tables in the database being converted. This message indicates successful completion of these updates.

**Action:** None required.

---

Error in slow altering a system table.

**Cause:** An internal error occurred while performing reversion.

**Action:** Contact Technical Support.

---

External conversion aborted due to incompatible sysmaster database.

**Cause:** The sysmaster database was not converted to the current database server version. A current sysmaster database is needed for external conversion to complete.

**Action:** Drop the sysmaster database and reboot the
database server. It will build a new **sysmaster** database and relaunch external conversion automatically.

### Messages: I-P

**Internal server error.**

**Cause:** An unexpected error occurred during database reversion.

**Action:** Contact Technical Support.

**Must drop long identifiers in table** *name* **in database** *name***

**Cause:** Identifiers greater than 18 bytes in length are not supported in the database server version to which you are reverting.

**Action:** Make sure that all long identifiers in the system are either dropped or renamed before you attempt reversion.

**Must drop new database** *(name)* **before attempting reversion.** *iserrno* **error_number***

**Cause:** The system contains a database that was created in a more recent version of the database server.

**Action:** Drop the new database and attempt reversion.

**Must drop new user defined statistics in database** *(name), iserrno* **number***

**Cause:** Some distributions in the sysdistrib system table use user-defined statistics. This feature is not supported in the version to which you are reverting.

**Action:** Ensure that no user-defined statistics are present or used in the system and then attempt reversion.

**ON-Bar conversion completed successfully.**

**Cause:** ON-Bar conversion completed successfully.

**Action:** None.

**ON-Bar conversion failed see /tmp/bar_conv.out.**

**Cause:** ON-Bar conversion failed.

**Action:** For failure details, see /tmp/bar_conv.out.

**ON-Bar reversion test started.**

**Cause:** ON-Bar reversion test script is now running.

**Action:** None.

**ON-Bar reversion test completed successfully.**

**Cause:** ON-Bar reversion test was completed successfully.

**Action:** None.

**'onload’ conversion completed successfully.**

**Cause:** Displayed in online.log at the successful completion of onload conversion.

**Action:** None required.

**'onload’ conversion failed. For details, look in $INFORMIXDIR/etc/onload.out.**

**Cause:** Conversion of the onload database failed.

**Action:** Find out the cause of failure from $INFORMIXDIR/etc/onload.out. Fix the problem before you reattempt conversion.

**...'onload’ reversion completed successfully.**

**Cause:** Printed in online.log at the successful completion of reversion.

**Action:** None required.

**...'onload’ reversion failed. For details, look in $INFORMIXDIR/etc/revload.out.**

**Cause:** Reversion of the onload database failed.

**Action:** Find the cause of failure in $INFORMIXDIR/etc/revload.out. Fix the problem before you reattempt reversion.

**'onload’ reversion test completed successfully.**
Cause: Printed in online.log if the onunload database is revertible.

Action: None required.

'onunload' reversion test start:

Cause: Printed in online.log at the beginning of onunload reversion testing.

Action: None required.

The onload database contains load/unload jobs referring to long table names, column names, or database names. These jobs will not work as expected until they are redefined.

Messages: R-W

...reverting 'onunload' database.

Cause: Printed in online.log at the beginning of onunload reversion.

Action: None required.

Reverting partition header from version 9.2.

Cause: The database server is reverting the partition header page to the old format that contains the physical address.

This message is optional verbose output that is logged only if you start oninit with the -v flag.

Action: None required.

Reverting partition header page address.

Cause: The database server is reverting the partition header page to the old format that contains the physical address.

This message is optional verbose output that is logged only if you start oninit with the -v flag.

Action: None required.

Reverting partition header pages status.

Cause: The database server is reverting the partition header pages to the old format. The status is identified as follows:

- started
- succeeded
- FAILED

Action: If reversion of the partition header pages started or succeeded, no action is required. If reversion of the partition header pages failed, restore from a tape backup, diagnose and fix the problem, and retry conversion.

Cause: Printed during onunload reversion testing if the onunload database contains references to long table names, column names, or database names. But the reversion will complete.

Action: Redefine the load and unload jobs in the onunload database that have references to long identifiers.

Reverting partition keys to pre 9.2.

Cause: The database server is reverting the partition keys to the pre-Version 9.2 format.

This message is optional verbose output that is logged only if you start oninit with the -v flag.

Action: None required.

Reverting partition name for databasename:tablename.

Cause: The database server is reverting the partition name for databasename:tablename.

This message is optional verbose output that is logged only if you start oninit with the -v flag.

Action: None required.

... reverting reserved pages.

Cause: The database server is reverting reserved pages.

Action: No action is required.

... reverting tables that underwent In-Place Alter.

Cause: The database server is reverting tables that underwent in-place alter.

Action: No action is required.

R-tree error message conversion completed successfully.

Cause: R-tree error message conversion was completed successfully.

Action: None required

R-tree error message conversion failed. (See /tmp/conrtree.out or %TMP%/conrtree.out)
Cause: R-tree error message conversion failed.
Action: See /tmp/conR-tree.out and /tmp/R-tree.databases.

---

R-tree error message conversion started.
Cause: R-tree error message conversion script is now running.
Action: None required.

---

Reversion cancelled.
Cause: The reversion process was cancelled because of errors encountered.
Action: Correct the cause of the errors, and restart reversion.

---

Reversion complete. Install IBM Informix database server version before restarting.
Cause: The reversion process was completed successfully.
Action: You must install the older database version.

---

Reversion of database name FAILED
Cause: Indicates the failure of reversion of the specified database.
Action: None required.

---

...reverting 'syscdr' database.
Cause: Printed in online.log at the beginning of Enterprise Replication reversion.
Action: None required.

---

...starting reversion of database name.
Cause: Indicates the start of actual reversion of the specified database.
Action: None required.

---

Unable to read reserved page chunk:offset - reserved_page.
Cause: Both disk pages in a given reserved page pair are bad. On the disk page, chunk represents the chunk number and offset represents the page offset for the chunk.
Action: Contact Technical Support.

---

WARNING: Target server version must have a certified Storage Manager installed after conversion/reversion and before bringing up server.
Cause: ON–Bar is being converted or reverted. The user must ensure that a storage manager, certified with the target database server version, is installed.
Action: None.

---

There is a semi-detached index in this table, which cannot be reverted. Drop this index, and retry reversion.
Cause: A semi-detached index on this table cannot be reverted.
Action: To see the list of all semi-detached indexes, refer to the database server message log. These indexes cannot be reverted. To continue reversion, drop these semi-detached indexes and retry reversion. If needed, you will need to re-create these indexes after reversion is complete.
Conversion and Reversion Messages for Enterprise Replication

Use the concdr.sh script on UNIX or the concdr.bat script on Windows to convert Enterprise Replication and the syscdr database to Version 10.0. Use the revcdr.sh script on UNIX or the revcdr.bat script on Windows to revert Enterprise Replication and the syscdr database to an earlier version. These scripts write conversion and reversion messages for Enterprise Replication to the following locations:

- Output of the concdr.sh or concdr.bat script, which is standard output by default
- concdr.out file
- Output of the revcdr.sh or revcdr.bat script, which is standard output by default
- revcdr.out file
- revtestcdr.out file

You can find the concdr.out, revcdr.out, and revtestcdr.out files in $INFORMIXDIR/etc on UNIX or %INFORMIXDIR%/etc on Windows. For more information on converting and reverting Enterprise Replication, see the IBM Informix Migration Guide.

---

**CDR reversion test completed successfully.**

**Cause:** The syscdr database is revertible.

**Action:** None required.

Prints the output of the revcdr.sh or revcdr.bat script to standard output.

---

**CDR reversion test failed; for details look in $INFORMIXDIR/etc/revtestcdr.out.**

**Cause:** Enterprise Replication is not revertible.

**Action:** For more information, look at the messages in revtestcdr.out. Fix the reported problem before you attempt reversion.

Prints the output of the revcdr.sh or revcdr.bat script to standard output.

---

**Enterprise Replication is not ready for conversion.**

The Control and TRG send queues should be empty for conversion/reversion to proceed.

**Cause:** There are elements in the control and Transaction Send Queue (also called TRG) send queues. The database server sends replicated data to the TRG queue before sending it to the target system.

**Action:** Wait for these queues to empty before you attempt either conversion or reversion. For more information, see the IBM Informix Dynamic Server Enterprise Replication Guide.

Prints this message to concdr.out during conversion or to revcdr.out during reversion.

---

**Enterprise Replication is not ready for conversion.**

The syscdr database should NOT contain old-style group definitions for conversion to succeed.

**Cause:** The syscdr database should not contain old-style group definitions for conversion to succeed.

**Action:** Use the cdr delete group command to delete the old-style groups before attempting conversion. For more information, see the IBM Informix Dynamic Server Enterprise Replication Guide.

Prints this message to concdr.out.

---

**Enterprise Replication should be in a stopped state for conversion/reversion to proceed.**

**Cause:** Enterprise Replication should be in a stopped state for conversion or reversion to proceed.

**Action:** Stop Enterprise Replication. For more information, see the IBM Informix Dynamic Server Enterprise Replication Guide.

Prints this message to concdr.out during conversion or to revcdr.out during reversion.

---

**Reversion of ‘syscdr’ failed; for details look in $INFORMIXDIR/etc/revcdr.out.**

**Cause:** The reversion of the syscdr database failed.

**Action:** Find the cause of failure in the revcdr.out file, then fix the problem before you attempt reversion.

Prints the output of the revcdr.sh or revcdr.bat script to standard output.

---

**Starting CDR reversion test...**

**Cause:** This message displays at the beginning of Enterprise Replication reversion testing.
Prints the output of the `revcdr.sh` or `revcdr.bat` script to standard output.

**Action:** None required.

---

**Starting 'syscdr' conversion...**

**Cause:** This message displays when you run the `concdr.sh` or `concdr.bat` script to convert the `syscdr` database to Version 10.0.

**Action:** None required.

Prints the output of the `concdr.sh` or `concdr.bat` script to standard output.

---

**Starting 'syscdr' reversion...**

**Cause:** This message displays when you run the `revcdr.sh` or `revcdr.bat` script to revert the `syscdr` database to an earlier version.

**Action:** None required.

Prints the output of the `revcdr.sh` or `revcdr.bat` script to standard output.

---

**'syscdr' conversion completed successfully.**

**Cause:** This message displays after you complete converting Enterprise Replication and the `syscdr` database to Version 10.0.

**Action:** None required.

Prints the output of the `concdr.sh` or `concdr.bat` script to standard output.

---

**'syscdr' conversion failed. For details, look in $INFORMIXDIR/etc/concdr.out.**

**Cause:** Conversion of the `syscdr` database failed.

**Action:** If conversion fails, resolve the problem reported in `concdr.out`. Restore the `syscdr` database from backup and reattempt conversion.

Prints the output of the `concdr.sh` or `concdr.bat` script to standard output.

---

**Syscdr should not contain replicates defined with the --floatiee option for reversion to succeed.**

**Cause:** Replicates have been defined with the `--floatiee` option. You cannot revert these replicates to the older version.

**Action:** Use the `cdr delete replicateset` command to delete replicates defined with the `--floatiee` option, then reattempt reversion.

Prints this message to `revtestcdr.out`.

---

Syscdr should NOT contain new replicate sets for reversion to succeed.

**Cause:** The new replicate sets in the `syscdr` database are not compatible with older versions.

**Action:** Use the `cdr delete replicateset` command to delete the replicate sets. Then rerun the `revcdr.sh` or `revcdr.bat` script to reattempt reversion.

Prints this message to `revtestcdr.out`.

---

**Dynamic Log Messages**
Dynamically added log file logid to DBspace dbspace_number.

Cause: The next active log file contains records of an open transaction. Whenever the database server adds a log dynamically, it logs this message. Example: Dynamically added log file 38 to DBspace 5.
Action: Complete the transaction as soon as possible.

Log file logid added to DBspace dbspace_number.

Cause: Whenever the administrator adds a log file manually, the database server logs this message. Example: Log file 97 added to DBspace 2.
Action: None required.

Log file number logid has been dropped from DBspace dbspace_number.

Cause: When you drop a newly-added log file, the database server logs this message. Example: Log file number 204 has been dropped from DBspace 17.
Action: None required.

Log file logid has been pre-dropped.

Cause: When you drop a used log file, it is marked as deleted (status D) and cannot be used again. After you perform a level-0 backup, the database server drops this log file and can reuse the space. Example: Log file 12 has been pre-dropped.
Action: To delete the log file, perform a level-0 backup of all storage spaces.

Pre-dropped log file number logid has been deleted from DBspace dbspace_number.

Cause: After a backup, the database server deletes a pre-dropped log file and logs this message. Example: Pre-dropped log file number 12 has been deleted from DBspace 3.
Action: None required.

Sbspace Metadata Messages

Allocated number pages to Metadata from chunk number.

Cause: The database server freed the specified number of pages from the reserved area and moved them to the metadata area of chunk number.
Action: None required.

Freeing reserved space from chunk number to Metadata.

Cause: The metadata area in chunk number is full. The database server is trying to free space from the reserved area to the metadata area.
Action: None required.

Allocated number pages to Userdata from chunk number.

Cause: The database server freed the specified number of pages from the reserved area and moved them to the user-data area of chunk number.

Freeing reserved space from chunk number to Userdata.

Cause: The user-data area in chunk number is full. The
database server is trying to free space from the reserved area to the user-data area.

**Action:** None required.

---

### Truncate Table Messages

The table cannot be truncated if it has an open cursor or dirty readers.

**Cause:** You must have exclusive access to the table.

**Action:** Wait for dirty readers to complete or close all the open cursors and reissue the TRUNCATE TABLE command.

---

The table cannot be truncated. It has at least one non-empty child table with referential constraints.

**Cause:** You cannot truncate a table if it has child tables with referential constraints and at least one row.

**Action:** Empty the child tables before you truncate this table.
Appendix F. Limits in IBM Informix Dynamic Server

The following sections list selected capacity limits and system defaults for IBM Informix Dynamic Server.

Limitations on UNIX Operating Systems

System-Level Parameter Limits (UNIX)

<table>
<thead>
<tr>
<th>System-Level Parameters</th>
<th>Maximum Capacity per Computer System</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Informix Dynamic Server systems per computer (Dependent on available system resources)</td>
<td>255</td>
</tr>
<tr>
<td>Maximum number of accessible remote sites</td>
<td>Machine specific</td>
</tr>
<tr>
<td>Maximum virtual shared memory segment (SHMVIRTSIZE)</td>
<td>2GB (32-bit platforms) or 4TB (64-bit platforms)</td>
</tr>
<tr>
<td>Maximum address space</td>
<td>1.7GB if boot.ini file not modified to 3GB 2.7GB if boot.ini file is modified to 3GB</td>
</tr>
</tbody>
</table>

Table-Level Parameter Limits (UNIX)

<table>
<thead>
<tr>
<th>Table-Level Parameters (based on 2K page size)</th>
<th>Maximum Capacity per Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data rows per fragment</td>
<td>4,277,659,295</td>
</tr>
<tr>
<td>Data pages per fragment</td>
<td>16,775,134</td>
</tr>
<tr>
<td>Data bytes per fragment (excludes Smart Large Objects (BLOB, CLOB) and Simple Large Objects (BYTE, TEXT) created in Blobspaces)</td>
<td>33,818,671,136</td>
</tr>
<tr>
<td>Binary Large Object BLOB/CLOB pages</td>
<td>4<em>2</em>40</td>
</tr>
<tr>
<td>Binary Large Objects TEXT/BYTE bytes</td>
<td>4<em>2</em>40</td>
</tr>
<tr>
<td>Row length</td>
<td>32,767</td>
</tr>
<tr>
<td>Number of columns</td>
<td>32K</td>
</tr>
<tr>
<td>Key parts per index</td>
<td>16</td>
</tr>
<tr>
<td>Columns per functional index</td>
<td>102 (for C UDRs) 341 (for SPL or Java UDRs)</td>
</tr>
<tr>
<td>Maximum bytes per index key (for a given page size):</td>
<td>2K page size = 387 4K page size = 796 8K page size = 1615 12K page size = 2435 16K page size = 3254</td>
</tr>
</tbody>
</table>
## Access Capabilities (UNIX)

<table>
<thead>
<tr>
<th>Access Capabilities</th>
<th>Maximum Capacity per System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum databases per Dynamic Server system</td>
<td>21 million</td>
</tr>
<tr>
<td>Maximum tables per Dynamic Server system</td>
<td>477,102,080</td>
</tr>
<tr>
<td>Maximum active users per Dynamic Server (minus the minimum number of system threads)</td>
<td>32K user threads</td>
</tr>
<tr>
<td>Maximum active users per database and table (also limited by the number of available locks, a tunable parameter)</td>
<td>32K user threads</td>
</tr>
<tr>
<td>Maximum number of open tables per Dynamic Server system</td>
<td>Dynamic allocation</td>
</tr>
<tr>
<td>Maximum number of open tables per user and join</td>
<td>Dynamic allocation</td>
</tr>
<tr>
<td>Maximum locks per Dynamic Server system and database</td>
<td>Dynamic allocation</td>
</tr>
<tr>
<td>Maximum number of page cleaners</td>
<td>128</td>
</tr>
<tr>
<td>Maximum number of recursive synonym mappings</td>
<td>16</td>
</tr>
<tr>
<td>Maximum number of tables locked with LOCK TABLE per user</td>
<td>32</td>
</tr>
<tr>
<td>Maximum number of cursors per user</td>
<td>Machine specific</td>
</tr>
<tr>
<td>Maximum Enterprise Replication transaction size</td>
<td>4 TB</td>
</tr>
<tr>
<td>Maximum chunk size</td>
<td>4 Terabytes</td>
</tr>
<tr>
<td>Maximum number of chunks</td>
<td>32,766</td>
</tr>
<tr>
<td>Maximum number of 2K pages per chunk</td>
<td>2 billion</td>
</tr>
<tr>
<td>Maximum number of open Simple Large Objects (applies only to TEXT and BYTE data types)</td>
<td>20</td>
</tr>
<tr>
<td>Maximum number of B-tree levels</td>
<td>20</td>
</tr>
<tr>
<td>Maximum amount of decision support memory</td>
<td>Machine specific</td>
</tr>
<tr>
<td>Maximum size of a Dynamic Server instance</td>
<td>8 PB</td>
</tr>
<tr>
<td>Utility support for large files</td>
<td>17 billion GB</td>
</tr>
</tbody>
</table>

## IBM Informix Dynamic Server System Defaults (UNIX)

<table>
<thead>
<tr>
<th>Table Lock Mode</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial extent size</td>
<td>8 pages</td>
</tr>
<tr>
<td>Next extent size</td>
<td>8 pages</td>
</tr>
<tr>
<td>Read-only isolation level (with database transactions)</td>
<td>Committed Read</td>
</tr>
<tr>
<td>Read-only isolation level (ANSI-compliant database)</td>
<td>Repeatable Read</td>
</tr>
</tbody>
</table>

## ON-Monitor Statistics (UNIX)

<table>
<thead>
<tr>
<th>Number of displayed user threads</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of displayed chunks</td>
<td>1000</td>
</tr>
<tr>
<td>Number of displayed dbspace</td>
<td>1000</td>
</tr>
</tbody>
</table>


| Number of displayed databases | 1000 |
| Number of displayed logical logs | 1000 |

**Limitations on Windows Operating Systems**

**System-Level Parameter Limits (Windows)**

<table>
<thead>
<tr>
<th>System-Level Parameters</th>
<th>Maximum Capacity per Computer System</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Informix Dynamic Server systems per computer (Dependent on available system resources)</td>
<td>255</td>
</tr>
<tr>
<td>Maximum number of accessible remote sites</td>
<td>Machine specific</td>
</tr>
<tr>
<td>Maximum virtual shared memory segment (SHMVIRTSIZE)</td>
<td>2GB (32-bit platforms) or 4TB (64-bit platforms)</td>
</tr>
<tr>
<td>Maximum address space</td>
<td>1.7GB if boot.ini file not modified to 3GB 2.7GB if boot.ini file is modified to 3GB</td>
</tr>
</tbody>
</table>

**Table-Level Parameter Limits (Windows)**

<table>
<thead>
<tr>
<th>Table-Level Parameters (based on 2K page size)</th>
<th>Maximum Capacity per Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data rows per fragment</td>
<td>4,277,659,295</td>
</tr>
<tr>
<td>Data pages per fragment</td>
<td>16,775,134</td>
</tr>
<tr>
<td>Data bytes per fragment (excludes Smart Large Objects (BLOB, CLOB) and Simple Large Objects (BYTE, TEXT) created in Blobspaces)</td>
<td>33,818,671,136</td>
</tr>
<tr>
<td>Binary Large Object BLOB/CLOB pages</td>
<td>4<em>2</em>40</td>
</tr>
<tr>
<td>Binary Large Objects TEXT/BYTE bytes</td>
<td>4<em>2</em>40</td>
</tr>
<tr>
<td>Row length</td>
<td>32,767</td>
</tr>
<tr>
<td>Number of columns</td>
<td>32K</td>
</tr>
<tr>
<td>Key parts per index</td>
<td>16</td>
</tr>
<tr>
<td>Columns per functional index</td>
<td>102 (for C UDRs) 341 (for SPL or Java UDRs)</td>
</tr>
<tr>
<td>Maximum bytes per index key (for a given page size):</td>
<td>2K page size = 387 4K page size = 796 8K page size = 1615 12K page size = 2435 16K page size = 3254</td>
</tr>
</tbody>
</table>

**Access Capabilities (Windows)**

<table>
<thead>
<tr>
<th>Access Capabilities</th>
<th>Maximum Capacity per System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum databases per Dynamic Server system</td>
<td>21 million</td>
</tr>
<tr>
<td>Maximum tables per Dynamic Server system</td>
<td>477,102,080</td>
</tr>
</tbody>
</table>

Appendix F. Limits in IBM Informix Dynamic Server F-3
<table>
<thead>
<tr>
<th>Access Capabilities</th>
<th>Maximum Capacity per System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum active users per Dynamic Server (minus the minimum number of system threads)</td>
<td>32K user threads</td>
</tr>
<tr>
<td>Maximum active users per database and table (also limited by the number of available locks, a tunable parameter)</td>
<td>32K user threads</td>
</tr>
<tr>
<td>Maximum number of open tables per Dynamic Server system</td>
<td>Dynamic allocation</td>
</tr>
<tr>
<td>Maximum number of open tables per user and join</td>
<td>Dynamic allocation</td>
</tr>
<tr>
<td>Maximum locks per Dynamic Server system and database</td>
<td>Dynamic allocation</td>
</tr>
<tr>
<td>Maximum number of page cleaners</td>
<td>128</td>
</tr>
<tr>
<td>Maximum number of recursive synonym mappings</td>
<td>16</td>
</tr>
<tr>
<td>Maximum number of tables locked with LOCK TABLE per user</td>
<td>32</td>
</tr>
<tr>
<td>Maximum number of cursors per user</td>
<td>Machine specific</td>
</tr>
<tr>
<td>Maximum Enterprise Replication transaction size</td>
<td>4 TB</td>
</tr>
<tr>
<td>Maximum chunk size</td>
<td>4 Terabytes</td>
</tr>
<tr>
<td>Maximum number of chunks</td>
<td>32,766</td>
</tr>
<tr>
<td>Maximum number of 2K pages per chunk</td>
<td>2 billion</td>
</tr>
<tr>
<td>Maximum number of open Simple Large Objects (applies only to TEXT and BYTE data types)</td>
<td>20</td>
</tr>
<tr>
<td>Maximum number of B-tree levels</td>
<td>20</td>
</tr>
<tr>
<td>Maximum amount of decision support memory</td>
<td>Machine specific</td>
</tr>
<tr>
<td>Maximum size of a Dynamic Server instance</td>
<td>8 PB</td>
</tr>
<tr>
<td>Utility support for large files</td>
<td>17 billion GB</td>
</tr>
</tbody>
</table>

**IBM Informix Dynamic Server System Defaults (Windows)**

<table>
<thead>
<tr>
<th>Table lock mode</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial extent size</td>
<td>8 pages</td>
</tr>
<tr>
<td>Next extent size</td>
<td>8 pages</td>
</tr>
<tr>
<td>Read-only isolation level (with database transactions)</td>
<td>Committed Read</td>
</tr>
<tr>
<td>Read-only isolation level (ANSI-compliant database)</td>
<td>Repeatable Read</td>
</tr>
</tbody>
</table>
Appendix G. Accessibility

IBM strives to provide products with usable access for everyone, regardless of age or ability.

Accessibility features for IBM Informix Dynamic Server

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility Features

The following list includes the major accessibility features in IBM Informix Dynamic Server. These features support:

- Keyboard-only operation.
- Interfaces that are commonly used by screen readers.
- The attachment of alternative input and output devices.

Tip: The IBM Informix Dynamic Server Information Center and its related publications are accessibility-enabled for the IBM Home Page Reader. You can operate all features using the keyboard instead of the mouse.

Keyboard Navigation

This product uses standard Microsoft Windows navigation keys.

Related Accessibility Information

IBM is committed to making our documentation accessible to persons with disabilities. Our publications are available in HTML format so that they can be accessed with assistive technology such as screen reader software. The syntax diagrams in our publications are available in dotted decimal format. For more information about the dotted decimal format, go to "Dotted Decimal Syntax Diagrams."

You can view the publications for IBM Informix Dynamic Server in Adobe Portable Document Format (PDF) using the Adobe Acrobat Reader.

IBM and Accessibility

See the IBM Accessibility Center at [http://www.ibm.com/able](http://www.ibm.com/able) for more information about the commitment that IBM has to accessibility.

Dotted Decimal Syntax Diagrams

The syntax diagrams in our publications are available in dotted decimal format, which is an accessible format that is available only if you are using a screen reader.

In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), the elements can appear on the same line, because they can be considered as a single compound syntax element.
Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read punctuation. All syntax elements that have the same dotted decimal number (for example, all syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, the word or symbol is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *

FILE with dotted decimal number 3 is read as 3 FILE. Format 3 FILE indicates that syntax element FILE repeats. Format 3 FILE indicates that syntax element FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol that provides information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this identifies a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should refer to a separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

? Specifies an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element (for example, 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

! Specifies a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicates that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines
2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP is applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP only applies to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

*Specifies a syntax element that can be repeated zero or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data-area, you know that you can include more than one data area or you can include none. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

Notes:
1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.

+Specifies a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times. For example, if you hear the line 6.1+ data-area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. As for the * symbol, you can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.
Notices

IBM may not offer the products, services, or features discussed in this document in all countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created
programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM’s future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM’s suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM’s application programming interfaces.
Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. (enter the year or years). All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

**Trademarks**

The following terms are used in one or more IBM Informix product publications and the terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

- AIX
- AnalystStudio
- C-ISAM
- Cloudscape
- DataBlade
- DB2
- DB2 Connect
- DB2 Universal Database
- Distributed Relational Database Architecture
- Domino
- DRDAA
- Dynamic Connect
- IBM
- Informix
- Lotus
- Lotus Notes
- MQSeries
- NetView
- Notes
- NUMA-Q
- OS/2
- OS/390
- OS/400
- Passport Advantage
- ProjectConsole
- Rational
- Rational Suite
- Rational Unified Process
- RedBack
- Red Brick
- RequisitePro
- RETAIN
- SQL/DS
- SystemBuilder
- Tivoli
- TME 10
- UniData
- UniVerse
- WebSphere
- z/OS

Adobe, Acrobat, Portable Document Format (PDF), and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.
Index

Special characters
-V option 6-2
-version option 6-2
.conf dbservename file A-4
.inf servicename file A-5
.informix file A-5
.inf os dbservename file
   defined A-6
   printing 15-23
   regenerating 11-19
.infxdirs file A-6
.jvprops file 1-53, A-7

Numerics
64-bit addressing
   buffer pool 1-20
   memory 1-86

A
AC_CONFIG environment variable 1-12
ac_config.std file 1-12, A-3
AC_DEBUG configuration parameter 1-12
AC_IXHRAR configuration parameter 1-12
AC_LTAPEBLOK configuration parameter 1-12
AC_LTAPEDEV configuration parameter 1-12
ac_msg.log file A-3
AC_MSGPATH configuration parameter 1-12
AC_SCHEMA configuration parameter 1-12
AC_STORAGE configuration parameter 1-12
AC_TAPEBLOCK configuration parameter 1-12
AC_TAPEDEV configuration parameter 1-13
AC_TIMEOUT configuration parameter 1-13
AC_VERBOSE configuration parameter 1-13
accessibility G-1
   keyboard G-1
   shortcut keys G-1
Accessibility
   dotted decimal format of syntax diagrams G-1
   syntax diagrams, reading in a screen reader G-1
ACCESTM tag 14-13
Active threads, printing 15-20
ADDCHK logical-log record 5-4
ADDDBS logical-log record 5-4
Adding
   CPU virtual processors 11-17, 11-18
ADDITEM logical-log record 5-4
ADDLOG logical-log record 5-4
ADMIN_MODE_USERS
   changing dynamically 11-21
ADMIN_MODE_USERS configuration parameter 1-13
ADMIN_USER_MODE_WITH_DBSA configuration parameter 1-13
Administration mode 11-14
Administrative API command
   ALTER LOGMODE 8-1
ADTERR configuration parameter 1-14
ADTMODE configuration parameter 1-14
ADTPATH configuration parameter 1-14
ADTSIZE configuration parameter 1-14
af.xxx file A-3
AFCRASH configuration parameter 1-53, A-7
AFDEBUG environment variable A-7
AFF_NPROC configuration parameter
   defined D-2
   VPCLASS D-2
AFF_SPROC configuration parameter
   defined D-2
   VPCLASS D-2
ALARMPROGRAM configuration parameter 1-14, C-1
Allocating unbuffered disk space 14-7, 14-8
ALLOCGENPG log record 5-4
ALLOW_NEWLINE configuration parameter 1-14
ALRM_ALL_EVENTS configuration parameter 1-15
ALTER LOGMODE
   Administrative API command 8-1
ALTERDONE log record 5-4
Alternate dbservename 1-26
ALTSPCLOGSNEW log record 5-4
ALTSPCLOGSOLD log record 5-4
archecker utility
   configuration parameters 1-12
Archiving
   after renaming spaces 14-25
Assertion failure
   af.xxx file A-3
   DUMPCNT configuration parameter 1-41
   DUMPCORE configuration parameter 1-41
   DUMPSTDOUT configuration parameter 1-42
   gcore file A-4
   shmem.xxx file A-8
Assertion-failed messages E-2
Asynchronous I/O
   cooked chunks D-7
   printing
      by virtual processor 15-22
      onstat -g ioa option 15-21
      onstat -g iof 15-22
      onstat -g log 15-22
Attribute
   configuration parameter 1-11
Audit records
   configuration parameters 1-14
   sysadtable 2-6
   sysaudtable 2-7
AUTO_AIOVPS
   changing dynamically 11-21
AUTO_AIOVPS configuration parameter 1-15
AUTO_CKPTS
   changing dynamically 11-21
AUTO_CKPTS configuration parameter 1-16
AUTO_LRU_TUNING
   changing dynamically 11-21
AUTO_LRU_TUNING configuration parameter 1-16
AUTO_REPREPARE configuration parameter 1-16
AVG_LO_SIZE tag 14-14

© Copyright IBM Corp. 1996, 2007
Blobspaces (continued)

- free-map page
  - defined 4-22
  - location in blobspace 4-21
  - role in blobpage logging 4-23
  - tracked by bitmap 4-23
- maximum number 14-6, 14-8, 14-17
- mirroring, ending 14-24
- mirroring, starting 14-22
- naming conventions 14-6
- page types 4-22
- restriction
  - dropping 14-20
  - simple-large-object storage 4-22

Block bitmap
- onstat -g nbm 15-22
- onstat -g rbm 15-23
- segment, nonresident 15-22
- segment, resident 15-23

Blocking
database server 11-5

BLOCKTIMEOUT configuration parameter 1-17

BMAP2TO4 logical-log record 5-5
BMAPPULL logical-log record 5-5
BSPADD logical-log record 5-5
BTCPYBCK logical-log record 5-5
BTMERGE logical-log record 5-5
BTSCANNER configuration parameter 1-18
BTHUFFL logical-log record 5-6
BTSPLIT logical-log record 5-6

Buffer pools 14-11
- 64-bit addressing 1-20
- adding a pool 13-4
- creating a pool 13-5
- smart large objects 1-21

Buffered
- disk space examples 14-4

BUFFERING tag
  - in -Df option 14-14

BUFFERPOOL configuration parameter 1-19

Buffers
- access-level flag bits 15-12
- page-type codes 15-11, 15-145
- read-ahead and page size 1-21

BUFFERS discontinued configuration parameter D-3

buffers field in the BUFFERPOOL configuration parameter 1-20

buildsmi script
- buildsmi.xxx file, defined A-4
- error log message E-8
- failure A-4
- initializing database server 2-2

buildsmi.xxx file A-4

C

data dictionary 15-21
  - SQL statements, printing 15-24

CDR logical-log record 5-6

CDR,DBSPACE configuration parameter 1-49
CDR,DSLOCKWAIT configuration parameter 1-49
CDR,ENV configuration parameter 1-49
CDR,EVALTHREADS configuration parameter 1-49
CDR,MAX_DYNAMIC_LOGS configuration parameter 1-49
CDR,NIFCOMPRESS configuration parameter 1-49
CDR,QDATA_SBSPACE configuration parameter 1-49
CDR_QHDR_DBSPACE configuration parameter 1-49
CDR_QUEUEMEM configuration parameter 1-49
CDR_SERIAL configuration parameter 1-49
CDR_SUPPRESS_ATSRISWARN configuration parameter 1-49
CHALLOC
  logical-log record 5-6
  record subtype (SBLOB) 5-15
Change
  physical log
    size and location 13-3
    text editor 13-4
Changing
  ONCONFIG in ISA 1-12
  sbspace attributes 14-17
  changing to 11-14
CHCOMBINE
  logical-log record 5-7
  record subtype, SBLOB 5-15
Checkpoint
  CKPTINTVL configuration parameter 1-23
  disabling I/O errors 1-71
  fast recovery performance 1-70
  statistics 2-8
CHFREE
  logical-log record 5-7
  record subtype, SBLOB 5-15
CHKADJUP log record 5-7
CHIPHYLOG log record 5-7
CHRESERV logical-log record 5-7
CHSPLIT
  logical-log record 5-7
  record subtype, SBLOB 5-15
Chunks
  changing mirroring status 14-26
  checking for overlap 7-12
  free list, checking with oncheck 7-5, 7-10
  free-list page 4-3, 4-4
  initial chunk of dbspace 4-2
  initial mirror offset 1-64
  maximum number 14-3
  monitoring 2-7
  structure
    additional dbspace chunk 4-3
    initial dbspace chunk 4-3
    mirror chunk 4-4
  using a symbolic link for the pathname 1-65, 1-81
CINDEX logical-log record 5-7
Cipher
  encryption 1-43
CKPOINT logical-log record 5-7
CKPTINTVL configuration parameter 1-23
CLEANERS configuration parameter 1-23
Client
  killing sessions (onmode -z) 11-23
  results of connection D-8
  specifying dbservername 1-27
  USEOSTIME configuration parameter 1-102
CLR logical-log record 5-7
CLUSIDX logical-log record 5-7
COARSELOCK log record 5-7
Cold restore
  number of recovery threads 1-69
CLOREPAI logical-log record 5-7
command_history table 3-5
Commands
  onstat -g cat 15-29

Commands (continued)
onstat -g ddr 15-33
onstat -g dss 15-39
onstat -g dsc 15-40
onstat -g grp 15-45
onstat -g nif 15-65
onstat -g que 15-76
onstat -g rcv 15-79
onstat -g rep 15-82
onstat -g rqm 15-83
onstat utility 15-83
COMMTR logical-log record 5-7
Compactness of index page 1-51
COMTAB logical-log record 5-8
COMWORK log record 5-8
concdr.out file E-26
concdr.sh script A-1, A-4, E-26
Configurable page size
  See BUFFERPOOL.
Configuration file
  displaying settings 1-5
    format 1-4
    preparing 1-5
    processing A-7
  warning about modifying onconfig.std 1-5, A-7
Configuration parameter
  ENCRYPT_MAC 1-46
  ENCRYPT_MACFILE 1-46
configuration parameters
  WSTATS 1-108
Configuration parameters
  AC_DEBUG 1-12
  AC_IxBAR 1-12
  AC_LTAPEBLOCK 1-12
  AC_LTAPEDEV 1-12
  AC_MSGPATH 1-23
  AC_SCHEMA 1-13
  AC_TIMEOUT 1-13
  AC_Verbose 1-13
  ADMIN_MODE_USERS 1-13
  ADMIN_USER_MODE_WITH DBSA 1-13
  ADTERR 1-14
  ADTMODE 1-14
  ADTPATH 1-14
  ADTSIZE 1-14
  AFCRASH 1-53, A-7
  AFF_NPROCS D-2
  AFF_SPROC D-2
  ALARMPROGRAM 1-14, C-1
  ALLOW_NEWLINE 1-14
  ALRM_ALL_EVENTS 1-15
  attributes 1-11
  AUTO_AIOVPS 1-15
  AUTO_CKPTS 1-16
  AUTO_LRU_TUNING 1-16
  AUTO_REPREPARE 1-16
  BAR_ACT_LOG 1-70
  BAR_BsAIB PATH 1-70
  BAR_DEBUG 1-70
  BAR_DEBUG_LOG 1-70
  BAR_HISTORY 1-70
  BAR_MAX_BACKUP 1-70
  BAR_NB_XPORT_COUNT 1-70
  BAR_PERFORMANCE 1-70

Index  X-3
Configuration parameters (continued)

- BAR_PROGRESS_FREQ 1-70
- BAR_RETRY 1-70
- BAR_XFER_BUF_SIZE 1-70
- BLOCKTIMEOUT 1-17
- BTSCANNER 1-18
- BUFFERPOOL 1-19
- CDR_HOTSPOT 1-49
- CDR_DBSIZE 1-49
- CDR_ENV 1-49
- CDR_EVALTHREADS 1-49
- CDR_MAX_DYNAMIC_LOGS 1-49
- CDR_NIFCOMPRESS 1-49
- CDR_QDATA_SUBSPACE 1-49
- CDR_QHDR_SUBSPACE 1-49
- CDR_QUEUEMEM 1-49
- CDR_SERIAL 1-49
- CDR_SUPPRESS_ATRISWARN 1-49
- CDR.Tx 1-11
- CKPTINTVL 1-23
- CLEANERS 1-23
- CONSOLE 1-24
- current default values 1-5
- DATASKIP 1-24
- DB_LIBRARY_PATH 1-25
- DB_CREATE_PERMISSION 1-25
- DBSERVERALIASES 1-25
- DBSERVERNAME 1-25
- DBSPACETEMP 1-28
- DD_HASHMAX 1-30
- DD_HASHSIZE 1-30
- DEADLOCK_TIMEOUT 1-31
- DEF_TABLE_LOCKMODE 1-31
- DIRECT_IO 1-32
- DIRECTIVES 1-32
- DISABLE_B162428_XA_FIX 1-33
- DRAUTO 1-34
- DRDA_COMMBUFFSIZE 1-33
- DRIDXAUTO 1-35
- DRINTV 1-35
- DRLOSTFOUND 1-36
- DRTIMEOUT 1-36
- DS_HASHSTART 1-37, 1-39
- DS_MAX_QUERIES 1-37, 11-10
- DS_MAX_SCANS 1-37, 11-11
- DS_NONPDQ_QUERY_MEM 1-38
- DS_POOLSIZE 1-37, 1-39
- DS_TOTAL_MEMORY 1-39, 11-10
- DUMPINT 1-41
- DUMPCODE 1-41
- DUMPPID 1-41
- DUMPFILE 1-42
- DUMPSHMEM 1-42
- DYNAMIC_LOGS 1-43
- ENCRYPT_CDR 1-49
- ENCRYPT_CIPHERS 1-43
- ENCRYPT_HDR 1-45
- ENCRYPT_MACFILE 1-50
- ENCRYPT_SMX 1-47
- EXPLAIN_STAT 1-50
- EXT_DIRECTIVES 1-50
- EXTSKADD 1-51
- FAST_RESTART_CKPT_FUZZYLOG 1-3
- FAST_RESTART_PHYSLOG 1-4
- FAST_POLL 1-51
- FILE_TYPE 1-51, 4-20
- HETERO_COMMIT 1-52
- hidden 1-5
- IFX_EXTEND_ROLE 1-52
- IFX_FOLDVIEW 1-52
- IMCLOG 1-63
- IMCTRANSPORTS 1-63
- IMCWORKERDELAY 1-63
- IMCWORKERTHREADS 1-63
- ISM_DATA_POOL 1-53, 1-71
- ISM_LOG_POOL 1-53, 1-71
- JVVMTHREAD 1-54
- JVCLASSPATH 1-54
- JVPDEBUG 1-53, A-7
- JVPHOME 1-53
- JVPJAVAHOME 1-54
- JVPJAVAHOME 1-54
- JVPJAVAVM 1-54
- JVPLOGFILE 1-53, A-7
- JVPPROFILE 1-53
- LISTEN_TIMEOUT 1-54
- LOG_INDEX_BUILDS 1-56
- LOGBUFF 1-55
- LOGFILES 1-56
- LOGSIZED 1-57
- LOGSIZED 1-57
- LTAP.BLUE 1-58
- LTAP.BLUE 1-58
- LTAP.BLUE 1-59
- LTAP.BLUE 1-59
- LTCHK 1-60
- LTCHW 1-60
- MAX_FILL_DATA_PAGES 1-61
- MAX_INCOMPLETE_CONNECTIONS 1-62
- MAX_PDQ_PRIORITY 1-62, 11-10
- MIRROR 1-64
- MIRROR_OFFSET 1-64
- MIRRORPATH 1-65
- MSGPATH 1-65
- MULTIPROCESSOR 1-66
- NETTYPE 1-67
- NUMAOP 1-7
- NUMCPUVPS 1-7
- OFF_RECVRY_THREADS 1-69
- ON_RECVRY_THREADS 1-70
- ON-Bar utility, types of 1-70
- ONDBACHEDOWN 1-71, 11-16
- ONLIDX_MAXMEM 1-72
- OPCACHE 1-72
- OPT_GOAL 1-73
- OPTCOMPID 1-73
- PC_HASHSIZE 1-75
- PC_HASHSIZE 1-75
- PHYSBUFF 1-75
- PHYSBUFF 1-75
- PHYSFILE 1-76
- PLOGOVERFLOW_PATH 1-77
- QSTATS 1-77
- RA_PAGES 1-78
- RA_THRESHOLD 1-78
- RESIDENT 1-79, 11-16
- RESTARTABLE_RESTORE 1-79
- ROOTOFFSET 1-80
- ROOTPATH 1-80, 1-81
- ROOTSIZE 1-81
- RTO_SERVER_RESTART 1-82
- SBSPACENAME 1-83, 1-97

X-4 IBM Informix Dynamic Server Administrator’s Reference
Configuration parameters (continued)

SBSPACETEMP 1-84, 14-13
SDS_ENABLE 1-84
SDS_PAGING 1-84
SDS_TEMPDBS 1-85
SDS_TIMEOUT 1-85
SECURITY_LOCALCONNECTION 1-86
SERVERNUM 1-86
setting decision-support with onmode 11-10
SHMADD 1-86
SHMBASE 1-87
SHMVIRT_ALLOCSEG 1-88
SHMVIRT_SIZE 1-89
SINGLE_CPU_VP 1-90
SQLTRACE 1-91
STACKSIZE 1-92
STAGEBLOB 1-92
STMT_CACHE 1-93
STMT_CACHE_HITS 1-93, 11-20
STMT_CACHE_NOLIMIT 1-94
STMT_CACHE_NUMPOOL 1-95
STMT_CACHE_SIZE 1-95
summary 1-6, 1-11
SYSLARMPROGRAM 1-95
SYSSBSPACENAME 1-97
TAPEBLK 1-97
TAPEDEV 1-98
TAPESIZE 1-99
TBLSPACE 1-100
TBLSPACE_STATS 1-100
TBTLBSECOND 1-101
TBTLBNEXT 1-101
TEMPTAB_NOLOG 1-100
TIMETOUT 1-101, 11-23
UNSECURE_ONSTAT 1-101
use
Data-replication screen 12-5
   Initialization screen 12-4
   PDQ screen 12-5
USELASTCOMMITTED 1-102
USEOSTIME 1-102
   viewing Enterprise Replication settings 15-30
VP_MEMORY_CACHE_KB 1-103
VPCLASS 1-54, 1-104, 11-18
CONSOLE configuration parameter 1-24
Conversion messages
   database server  E-21, E-25
   Enterprise Replication  E-26, E-27
Core dump
   contained in core file  A-4
   DUMPCORE parameter  1-41
core.pid.cnt file 1-42
CPU
time tabulated 15-130
CPU virtual processor
   binding  D-2
   SINGLE_CPU_VP parameter 1-90
Create Dbspace with On-Monitor  12-5
CREATE FUNCTION statement 1-106
CREATE INDEX statement
   using FILLFACTOR 1-51
CREATE record subtype (SBLOB) 5-15
Creating
   buffer pool 13-5
curl field 15-143

D

d daemon.log A-6
Data
   sync threads 15-21
Data distributions
   sbspaces 1-97
Data pages
   number to read ahead 1-78
   oncheck -cd 7-9
   oncheck -cD 7-9
   oncheck -cd and -cD 7-5
Data replication
   ENCRYPT_CIPHERS 1-43
   ENCRYPT_HDR 1-45
   ENCRYPT_MAC 1-46
   ENCRYPT_MACFILE 1-46
   ENCRYPT_SMX 1-47
   ENCRYPT_SWITCH 1-48
   flush interval 1-35
   information in sysdri table 2-11
   lost-and-found file 1-36
   onstat -g dri statistics 15-21
   wait time for response 1-36
Data row
   big-remainder page 4-15
   forward pointer 4-13
   home page 4-13, 4-15
   locating the row 4-13
   rowid 4-13
   storage strategies 4-13
   storing data on a page 4-14
   TEXT and BYTE data descriptor 4-22
Data-dictionary cache 1-30
Data-distribution cache
   onstat -g disc 15-21
   specifying
      entries 1-39
      hash buckets 1-37
Database servers
   blocking 11-5
   bringing online from quiescent mode 11-13, 11-14
   name 1-27
   parallel database query 11-17
   quiescent mode 11-13, 11-14
   remote 2-23
   restarting 1-11
   shutting down 11-13, 11-14
   unblocking 11-5
   using onstat -g pos 15-23
Database tblspace
   entries 4-7
   location in root dbspace 4-2, 4-7
   relation to systable 4-27
   structure and function 4-7
   tblspace number 4-7
Databases
   effect of creation 4-26
   Enterprise Replication log reader  15-20
   locale, in sysdbslocale table 2-10
   owner, in sysmaster database 2-10
   sysdatabases table 2-10
   DATASKIP configuration parameter
      defined 1-24
      using onspaces -f 14-21
   DB_LIBRARY_PATH configuration parameter 1-25
   DBCREATE_PERMISSION configuration parameter 1-25
DBSERVERALISES configuration parameter
  defined  1-26
  using onmode -d 11-7
DBSERVERNAME configuration parameter
  defined  1-27
  using onmode -d 11-7
dbspaces
  adding chunk  14-3
  blobpage structure  4-21
  creating
    with onspaces  14-8
  dropping
    chunk  14-19
    with onspaces  14-20
  ending mirroring  14-24
  list of structures contained  4-3
  maximum number  14-6, 14-8, 14-17
  modifying with onspaces  14-21
  monitoring with SMI  2-10
  naming conventions  14-8
  root name  1-80
  simple-large-object storage  4-22
  starting mirroring  14-22
  storage  4-2
  structure
    additional dbspace chunk  4-3
    chunk free-list page  4-4
dbspace  4-2, 4-3
  mirror chunk  4-4
  nonroot dbspace  4-3
tblspace tblspace  4-5
DBSPACETEMP configuration parameter  1-28
DD_HASHMAX configuration parameter  1-30
DD_HASHSIZE configuration parameter  1-30
Deadlock  1-31
DEADLOCK_TIMEOUT configuration parameter  1-31
Decision-support queries
  DS_MAX_QUERIES configuration parameter  1-37
  DS_TOTAL_MEMORY configuration parameter  1-39
  gate information  15-63
  gate numbers  15-64
  MAX_PDQ_PRIORITY configuration parameter  1-63
  setting parameters with onmode  11-10
DEF_TABLE_LOCKMODE configuration parameter  1-31
Default configuration file  1-5, A-8
DELETE
  logical-log record  5-8
  record subtype, GBLOB  5-15
Delete table cleaner  15-21
DELITEM logical-log record  5-8
DELETE logical-log record  5-8
Descriptor, TEXT and BYTE data  4-22
Diagnotic
  using onmode  B-1
DINDEX logical-log record  5-8
DIRECT_IO configuration parameter  1-32
DIRECTIVES configuration parameter  1-32
Disabilities, visual
  reading syntax diagrams  G-1
disability  G-1
DISABLE_B162428_XA_FIX configuration parameter  1-33
Disabling SQL statement cache  11-11
Disk I/O
  buffers  1-21
  PDQ resources  1-63
Disk page
  page compression  4-15
Disk page (continued)
  storing data on a page  4-14
    structure
      blobspace blobpage  4-8
dbspace page  4-12
  types of pages in an extent  4-8, 4-9
Disk space
  allocating
    for system catalogs  4-26
    when a database is created  4-26
    when a table is created  4-27
  chunk free-list page  4-4
  initializing (oninit -i)  9-1
  list of structures  4-2
  maximum chunk size  14-4, 14-7, 14-10, 14-12, 14-17
  page compression  4-15
  tracking available space in
    blobspace  4-22
    chunk  4-4
Distributed transactions
  killing  11-23
Dotted decimal format of syntax diagrams  G-1
drl.lostfound file  1-35
DRAUTO configuration parameter  1-34
DRDA_COMMBUFFSIZE  1-33
DRIDXAUTO configuration parameter  1-35
DRINTERVAL configuration parameter  1-35
DRLOSTFOUND configuration parameter  1-36
DROP DISTRIBUTIONS keyword 1-97
DRPDBS logical-log record  5-8
DRPCHK logical-log record  5-8
DRPLOG logical-log record  5-8
DRTIMEOUT configuration parameter  1-36
DS_HASHSIZE configuration parameter  1-37, 1-39
DS_MAX_QUERIES
  changing dynamically  11-21
  DS_MAX_QUERIES configuration parameter
    changing value  11-10
    defined  1-37
DS_MAX_SCANS
  changing dynamically  11-21
  DS_MAX_SCANS configuration parameter
    changing value  11-11
    defined  1-38
DS_NONPDQ_QUERY_MEM
  changing dynamically  11-21
  DS_NONPDQ_QUERY_MEM configuration parameter
    1-38
  DS_POOLSIZE configuration parameter  1-37, 1-39
  DS_TOTAL_MEMORY
    changing dynamically  11-21
  DS_TOTAL_MEMORY configuration parameter
    changing value  11-10
    defined  1-40
DUMPCNT configuration parameter  1-41
DUMPCORE configuration parameter  1-41
DUMPDIR configuration parameter
  af.xxx assertion failure file  A-3
  defined  1-41
gcore file  A-4
  shmem file  A-8
  DUMPCORE configuration parameter  1-42
  Dumping stack of thread  15-24
  DUMPshm configuration parameter  1-42
  Dynamic libraries, loaded  15-21
  Dynamic log messages  E-28
  DYNAMIC_LOGS configuration parameter  1-43
E

Emergency boot file A-4
Enabling SQL statement cache 11-11
ENCRYPT_CDR configuration parameter 1-49
ENCRYPT_CIPHES configuration parameter 1-43, 1-49
ENCRYPT_HDR configuration parameter 1-45
ENCRYPT_MAC configuration parameter 1-46, 1-49
ENCRYPT_MACFILE configuration parameter 1-46, 1-50
ENCRYPT_SMX configuration parameter 1-47
ENCRYPT_SWITCH configuration parameter 1-48, 1-50
Encryption
- cipher renegotiation 1-48
- high-availability data replication 1-45
- MAC files, specifying 1-46
- message authentication code generation 1-46
- specifying ciphers and modes 1-43
ENDTRANS logical-log record 5-8
Enterprise Replication
- CDR log record 5-6
- configuration parameters 1-49
- messages E-26, E-27
- onstat commands
  - g cat 15-20
  - g ddr 15-20
  - g dss 15-21
  - g dtc 15-21
  - g gtp 15-21
  - g nif 15-22
  - g que 15-23
  - g rcv 15-23
  - g rep 15-23
  - g rqn 15-23
- queued events 15-23
- renaming spaces 14-25
- RQM statistics 15-23
Environment configuration file A-5
Environment variables
- AC_CONFIG 1-12
- AFDEBUG A-7
- defined 1-12
- IFX_DEF_TABLE_LOCKMODE 1-31
- IFX_DIRECTIVES 1-32, 1-50
- IFX_XASTDCOMPLIANCE_XAEND 1-33
- IMCADMIN 1-63
- IMCONFIG 1-63
- IMCSERVER 1-63
- INFORMIXDIR 1-63
- INFORMIXOPCACHE 1-72
- INFORMIXSERVER 1-27, 1-63, 11-19
- INFORMIXSQLHOSTS 1-63
- ONCONFIG
  - defined A-7
  - onstat -c 15-13
  - setting 1-5
- OPTCOMPD 1-73
- SERVER_LOCALE 15-42
- STMT_CACHE 1-93, 11-11
- viewing Enterprise Replication settings 15-30
- ERASE logical-log record 5-8
Event alarm
- ALARMPROGRAM parameter 1-14, C-1
- automatic log backup 1-14, C-1
- class ID parameter C-3
- class message parameter C-3
- creating 2-4
- defined C-1
- event severity codes C-2

Event alarm (continued)
- exit code C-2
- using ex_alarm.sh C-1
- writing your own script C-1
- Event severity codes C-2
- ex_alarm.sh script 1-14, C-1
- Exclusive-access, high-water mark 1-60
- EXE.sessionid.threadid 15-22
- EXIT_STATUS exit code C-2
- Exit, return codes on 15-146
- EXPLAIN_STAT
  - changing dynamically 11-21
- EXPLAIN_STAT configuration parameter 1-50
- EXT_DIRECTIVES configuration parameter 1-50
- EXTEND record subtype, SBLOB 5-15
- EXTENT_SIZE tag 14-15
- Extents
  - automatic doubling of size 4-11
  - default size 4-8
  - disk page types 4-8, 4-9
  - merging 4-11
  - next-extent, allocating 4-11, 4-12
  - procedure for allocating 4-11
  - size
    - index fragments 4-8
    - initial extent 4-8
    - next extent 4-11
  - structure 4-8
  - sysextent table 2-12
  - External backup
    - commands 11-5
  - EXTSHMADD configuration parameter 1-51
- Extspace
  - creating 14-17
  - dropping 14-20
  - naming conventions 14-17
  - specifying location 14-17
  - sysextspaces table 2-13

F

FAST_RESTART_CKPT_FUZZYLOG configuration parameter D-3
FAST_RESTART_PHYSLOG configuration parameter D-4
FASTPOLL configuration parameter
- defined 1-51
Files
- .conf.dbservername A-4
- .inf.servicename A-5
- .informix A-5
- .infos.dbservername A-6
- .infdirs A-6
- .jyprops 1-53, A-7
- ac_config,std A-3
- ac_msg.log A-3
- archecker configuration file A-3
- bar_act.log A-4
- buildsmi.xxx A-4
- core.pid.cnt 1-42, A-4
- default configuration file A-8
- dr.lostfound 1-35
- gcore 1-41, A-4
- informix.rc environment file A-5
- INFORMIXTMP directory A-5
- ISM logs A-6
- ISMVersion A-6
- JVM_vpid 1-53, A-7
Files (continued)
JVPLOG 1-53, A-7
oncfg* A-8
ONCONFIG A-8
private environment hash file A-5
shm.pid.cnt 1-42
shm.xxx A-8
sm_versions A-8
summary of database server files A-1
VP.servername.nnx A-9
xbasa.messages A-9
FILLFACTOR configuration parameter
control how indexes fill 4-20
defined 1-51
Flushing
data-replication buffer 1-35
SQL statement cache 11-11
Force option, onspaces 14-21
Forced residency
starting and ending with onmode 11-16
Formula
quantum of memory 15-61
Forward pointer
blobspace blobpage 4-23
dbspace storage of simple large objects 4-22
defined 4-13
Fragment
allocated by use 15-25
index 4-8
internal structure of tables 4-15
rows 4-14
table, using primary keys 4-14
turning DATASKIP ON or OFF for 14-21
warning returned when skipped during query 1-24
FREE_RE logical-log record 5-9
Free-map page, blobspace 4-22
Freeing
blobpages 15-17
unused memory segments 11-12
Functional index 4-20

G
Gateway transactions 1-52
gcore
file A-4
utility 1-41, 1-42
General Page Manager 15-129
Global catalog
Enterprise Replication 15-20
Global transactions
using onstat -G 15-116
using onstat -x 15-144
Grouper, Enterprise Replication 15-21

H
Hash buckets
data-dictionary cache 1-30
data-distribution cache
specifying hash buckets 1-37
HDELETE logical-log record 5-9
HDRUPD record subtype, SBLOB 5-15
HETERO_COMMIT configuration parameter 1-52
Heterogeneous-commit transactions 1-52
HEURTX logical-log record 5-9
High-water mark, transaction 1-60
HINSERT logical-log record 5-9
Home page 4-13, 4-15
HUPAFF logical-log record 5-9
HUPBEF logical-log record 5-9
HUPDATE logical-log record 5-9

I
I/O
lightweight 14-14
IBM Informix Server Administrator
adding or dropping logs 1-56, 1-57
defined 6-3
setting ONCONFIG parameters 1-5, 1-12
IBM Informix STAR queries 15-142
IBM Informix Storage Manager
catalog A-6
ISMversion file A-6
logs A-6
sm_versions file A-8
Identifier, defined 1-27
IDXFLAGS logical-log record 5-10
ifx_allow_newline() routine 1-15
IFX_DEF_TABLE_LOCKMODE environment variable 1-31
IFX_DIRECTIVES environment variable 1-32, 1-50
IFX_EXTEND_ROLE configuration parameter 1-52
IFX_FOLDVIEW configuration parameter 1-53
ifx_lo_specset_estbytes function 14-15
ifx_lo_stat function 14-19
IFX_XASTDCOMPLIANCE_XAEND environment
variable 1-33
illsrra.xx file A-5
IMCADMIN environment variable 1-63
IMCCONFIG environment variable 1-63
IMCLOG configuration parameter 1-63
IMCSERVER environment variable 1-63
IMCTRANSPORTS configuration parameter 1-63
IMCWORKERDELAY configuration parameter 1-63
IMCWORKERTHREADS configuration parameter 1-63
Index
branch node 4-16
duplicate key values 4-19
defined 4-20
functional 4-20
how created and filled 4-17
item described 4-17
key value locking 4-19
leaf node 4-16
reuse of freed pages 4-20
root node 4-16
structure of B-tree 4-16
Index item
calculating the length of 4-20
defined 4-17
Index page
compactness 1-51
creation of first 4-17
effect of creation 4-17
structure 4-16
Index page logging 15-22
Information
SQL profile 2-25
informix.rc environment file A-5
INFORMIXDIR environment variable 1-63
INFORMIXOPCACHE environment variable 1-72
INFORMIXSERVER environment variable 1-27, 1-63, 11-19
Limits (continued)
virtual processors 11-17
Linking, name of root dbspace 1-81
LISTEN_TIMEOUT
changing dynamically 11-21
LISTEN_TIMEOUT configuration parameter 1-54
Locales xvi
Location, extspace 14-17
Lock
buffer-access-level flag bits 15-12
information in syslocks table 2-15
key-value 4-19
maximum time to acquire 1-31
monitoring with onstat -k 15-6, 15-121
multiprocessor 1-66
oncheck options 7-2
type codes 15-121
Lock mode, page or row 1-31
LOCK_MODE tag 14-14
Locked mutexes 15-22
LOCKS configuration parameter 1-55
Log position 15-143
log_full scripts 1-14, C-1
LOG_INDEX_BUILDS configuration parameter 1-56
LOGBUFF configuration parameter 1-55
LOGFILES configuration parameter 1-56
Logging
blobspace free-map page 4-23
flags for mode 4-8
LOGGING tag 14-15
Logical log
adding files 13-2
backup
alarm triggered 1-14, C-1
checking consistency 7-12
dropping files 13-3
file
created during initialization 1-56
displaying contents 10-1
log position 15-143
moving 13-3
reading the log file 10-1
size 1-57
switching with onmode 11-15
files
maximum number 13-2
minimum number 13-3
in root dbspace 4-2
log position 15-143
maximum size 15-143
monitoring with SMI 2-16
onparams
adding files 13-2
dropping files 13-3
record
additional columns 5-3
checkpoint 5-2
displaying 10-1
distributed transactions 5-2
DROP TABLE operation 5-1
generated by rollback 5-2
grouper information 15-21
header columns 5-3
types 5-3, 5-14
specifying file size 13-2
Logical page contents
displaying with oncheck 7-16

INFORMIXSQLHOSTS environment variable 1-63
INFORMIXTMP directory A-5, A-6
Initializing
disk structures 1-11, 4-2
shared memory 1-11
INSERT logical-log record 5-10
InstallServer.log file A-6
Interrupt signal 6-1
Interval
checkpoint 1-23
ISA.
See IBM Informix Server Administrator.
ISAM calls tabulated 15-129
ISM_DATA_POOL configuration parameter 1-53, 1-71
ISM_LOG_POOL configuration parameter 1-53, 1-71
ism_startup command A-8
ISM.
SeeIBM Informix Storage Manager. A-6
ISVERSION file A-6
ISOSP_COMMIT log record 5-10

J
Java configuration parameters 1-53
Java virtual processor 1-54
JDBC configuration parameter 1-53
JDKVERSION configuration parameter 1-53
JVM.pid file 1-53, A-7
JVMTHREAD configuration parameter 1-54
JVCLASSPATH configuration parameter 1-54
JVDEBUG configuration parameter 1-53, A-7
JVMHOME configuration parameter 1-53
JVPJAVAHOME configuration parameter 1-54
JVPJAVA configuration parameter 1-54
JVPJAVAVM configuration parameter 1-54
JVPLOG file 1-53, A-7
JVPLOGFILE configuration parameter 1-53, A-7
JVPJVMFILE configuration parameter 1-53

K
Key value
checking order with oncheck 7-5, 7-10
duplicates 4-19
locking 4-19
Key-only
inserting entries 1-94, 11-20
Killing a session 11-23

L
Large chunk mode 11-5
Latch
displaying with onstat -s 15-7, 15-136
identifying the resource controlled 15-137
LBU_PRESERVE discontinued configuration parameter D-5
LCKLVL logical-log record 5-10
LG_ADDBPOOL logical-log record 5-10
LG_CDINDEX log record 5-6
Library
dynamic 15-21
Licensed users, maximum allowed E-11
Light scans 15-22
Lightweight I/O 14-14
Limits
SQL statement cache size 1-94, 11-20

Index X-9
Logical recovery, number of threads 1-70
Logical-log buffer and LOGBUFF configuration
parameter 1-55
logmessage table  E-2
logpbf field  15-143
LOGSIZE configuration parameter  1-57
LOGSMAX discontinued configuration parameter  D-5
Long transaction
  high-water mark  1-60
LTEXEHWM  1-60
LTXHWM  1-60
Longspins 15-24
Loosely-coupled mode 15-144
LRU
  changing dynamically  11-21
LRU queues
    displaying with onstat -R  15-7, 15-134
    FLRU queues  15-134
    MLRU queues  15-134
    modified pages, percentage  1-22
    LRU_MAX_DIRTY discontinued configuration parameter  D-5
    lru_max_dirty field in the BUFFERPOOL configuration
      parameter  1-22
    LRU_MIN_DIRTY discontinued configuration parameter  D-6
    lru_min_dirty field in the BUFFERPOOL configuration
      parameter  1-22
    LRUS discontinued configuration parameter  D-6
    lrus field in the BUFFERPOOL configuration parameter  1-20
    LTAPEBLK configuration parameter  1-58
    LTAPEDEV configuration parameter  1-58
    LTAPESIZE configuration parameter  1-59
    LTEXEHWM configuration parameter  1-60
    LTXHWM configuration parameter  1-60
Memory
MAX_FILL_DATA_PAGES configuration parameter  1-61
MAX_INCOMPLETE_CONNECTIONS
  changing dynamically  11-21
MAX_INCOMPLETE_CONNECTIONS configuration
parameter  1-62
MAX_PDQ_PRIORITY
  changing dynamically  11-21
MAX_PDQ_PRIORITY configuration parameter
  changing value  11-10
  defined  1-62
MaxConnect
  configuration parameters  1-63
  DBSERVERALIAS configuration parameter  1-26
  DBSERVERNAME configuration parameter  1-27
  environment variables  1-63
  NETTYPE configuration parameter  1-69
  network statistics  15-23
  onstat -g inc  15-21
  onstat -g nta  15-23
Maximum number
  chunks  14-3
  storage spaces  14-6, 14-8, 14-12, 14-17
Maximum user connections  E-11
Memory
  allocated fragments  15-20
  freeing unused segments  11-12
  pools, SQL statement cache  1-95
  quantum allocated by MGM  15-61, 15-63
  specifying size of  1-72
  SQL statements  15-25
  statistics for pools  15-22
Memory Grant Manager  15-22
  monitoring resources  15-61
Memory information
  SMI tables  2-17, 2-21
Message authentication code files  1-46
Message log
  alphabetical listing of messages  E-1
  categories of messages  E-2
  defined A-7, E-1
  displaying with onstat -m  15-6, 15-125
  event alarms  C-2
  location  1-65
  viewing messages  E-1
Messages
  A-B  E-2, E-3
  assertion-failed  E-2
  C  E-3, E-7
  changing sbspace minimum extent size  14-15
  conversion and reversion  E-21, E-25
  D-E-F  E-7, E-9
  dynamic log  E-28
  Enterprise Replication  E-26, E-27
  G-H-I  E-9, E-10
  in-place ALTER TABLE  E-8
  J-K-L-M  E-10, E-12
  N-O-P  E-12, E-15
  onspaces  E-14
  Q-R-S  E-15, E-18
  sbspace metadata  E-28, E-29
  symbols  E-20, E-21
  T-U-V  E-18, E-20
  truncate table  E-29
  turning off smart-large-object logging  14-15
  W-X-Y-Z  E-20
Metadata
  area, structure  4-24
  checking with oncheck  7-2
  creating  14-12
  messages  E-28, E-29
  size  14-12, 14-14
  specifying offset  4-23, 14-12
  specifying size  14-12
  temporary sbspace  1-84
  MGM
    information  2-17
    mi_lo_decrefcount() function  14-19
    mi_lo_increfcount() function  14-19
    mi_lo_specset_estbytes() function  14-15
    mi_lo_stat function()  14-19
Microsoft Transaction Server
  defined  15-142
  onstat -x output  15-144
MIN_EXT_SIZE tag  14-15
Mirror chunk, structure  4-4
MIRROR configuration parameter  1-64
Mirroring
  changing chunk status  14-26
  enable flag  1-64
  initial chunk  1-65
  starting  14-22
  stopping  14-24
MIRROR_OFFSET configuration parameter  1-64
MIRRORPATH configuration parameter  1-65
Modes
  encryption  1-43

X-10  IBM Informix Dynamic Server Administrator's Reference
Modified pages specifying percentage LRU queue 13-5
Monitoring display environment variables 15-21, 15-41
distributed queries 15-142 licensed user connections E-11
MGM resources 15-61
Moving logical-log files 13-3
MSGPATH configuration parameter 1-65
Multiprocessor computer
  AFF_SPROC configuration parameter D-2
  processor affinity 1-107
MULTIPROCESSOR configuration parameter 1-66
Mutex
  locked 15-22
  onstat -g rwm 15-23
  with waiters 15-25
MVIDXND logical-log record 5-10

N
Name
  blobspace 14-6
dbspace 14-8
extspace 14-17
sbspaces 14-12
NETTYPE configuration parameter defined 1-67
tuning example 1-67
Network information
  SMI tables 2-17, 2-18
Network statistics
  Enterprise Replication 15-22
  shared memory 15-23
Newline character, quoted strings 1-14
NEXT_SIZE tag 14-15
Next-extent
  allocation 4-11
  allocation strategy 4-12
doubling of size 4-11
  initial size 4-8
  nonfragmented table 4-9
no_log scripts 1-14
NOAGE configuration parameter
  VPCLASS 1-105
NOAGE discontinued configuration parameter D-7
Node, index branch
  creating 4-18
  defined 4-16
  what points to 4-18
checking horizontal and vertical nodes 7-5, 7-10
  defined 4-17
leaf
  contents 4-18
  defined 4-16
 pointer 4-18
root node
  creating 4-17
  defined 4-16
  when fills 4-17
types 4-16
Non-default page size
  physical log 13-4
Nonresident segment, printing 15-22
NUMAIOVPS configuration parameter defined D-7
  VPCLASS 1-105
Number of page-cleaner threads 1-23
NUMCPUVPS configuration parameter defined D-8
  VPCLASS 1-105
On
Object Explorer 6-4
OFF_RECVRY_THREADS configuration parameter 1-69
Offset
  mirrored chunk 14-12
  size 14-4, 14-7, 14-10, 14-12, 14-17
ON_RECVRY_THREADS configuration parameter 1-70
ON-Bar utility
  activity log A-4
  configuration parameters 1-70
  emergency boot files A-4
  sm_versions file A-8
  system tables 2-6
  xbsa.messages log A-9
On-Monitor Create Dbsapce name 12-5
ON-Monitor utility
  Archive menu options 12-3
  changing
    database server mode 11-15
    parameter values 1-12
  Dbspaces menu options 12-2
  Diagnostics screen 12-5
  displaying system page size 1-22
  executing shell commands 12-2
  Force-Ckpt menu options 12-3
  help 12-1
  Initialization screen 12-4
  Logical-Logs menu options 12-3
  Mode menu options 12-3
  navigating 12-1
  Parameters menu options 12-2
  Performance screen 12-4
  Shared-memory screen 12-4
  Status menu options 12-2
  using 12-1
oncfg file
  and onspaces 14-2
  defined A-8
oncheck utility
  check-and-repair 7-1
  defined 7-1
  display reserved, physical-log, and logical-log pages 4-3
  list of functions 7-2
  locking 7-3
  option descriptions 7-4
  options
    -cc 7-8
    -cd 7-9
    -cD 7-9
    -ce 7-5, 7-10
    -ci 7-10
    -cI 7-10
    -cr 7-12
    -cR 7-12
    -cs 7-12
    -cS 7-12
    -n 7-3, 7-5
    -pB 7-13

Index X-11
oncheck utility (continued)
  options (continued)
    -pc 7-8
    -pd 7-13
    -PD 7-13
    -pe 4-2, 7-10
    -pk 7-15
    -PK 7-15
    -pl 7-15
    -PL 7-15
    -pp 7-16
    -PP 7-16
    -pr 1-5, 4-3, 7-18
    -PR 4-3, 7-18
    -ps 7-12
    -PS 7-12
    -pt 7-18
    -PT 7-18
    -u 7-20
    -x 7-20
    -y 7-3, 7-7
overview of functionality 7-1
suppressing messages 7-7
syntax 7-4
ONCONFIG configuration file
changing parameter values 1-11
changing with ISA 1-12
conventions 1-6
defined A-8
displaying 1-5, 15-6, 15-13
format 1-4
preparing 1-5
specifying hidden parameters 1-5
templates 1-4
white space 1-4
ONCONFIG environment variable
ONCONFIG file A-7
setting 1-5
using onstat -c 15-13
onconfig.std file
default value 1-11
defined 1-4
hidden parameters 1-5
printing 1-5
ondblog utility 8-1
BAR_ACT_LOG file 8-1
defined 8-1
ONDBSPACEDOWN configuration parameter
defined 1-71
overriding WAIT mode 11-16
oninit utility
option descriptions 9-3
options
  -i 9-3
  -p 9-2
  -s 9-2, 9-3
  -S 9-2
  -U 9-2
starting database server 9-1
temporary tables 9-2
ONLIDX_MAXMEM
changing dynamically 11-21
ONLIDX_MAXMEM configuration parameter 1-72
onlog utility
defined 10-1
filters for logical-log records
displaying 10-3
onlog utility (continued)
  filters for logical-log records (continued)
    reading 5-6, 10-2
options
  -b 10-3
  -d 10-3
  -l 5-6, 10-4
  -n 10-3
  -q 10-2
  -t 10-4
  -u 10-4
  -x 10-4
onmode utility
adding
  shared-memory segment 11-4
  virtual processors 11-17
administration mode 11-14
blocking the database server 11-5
changing
  database server mode 11-12, 11-14, 11-15
  DS_MAX_QUERIES 11-10
  DS_MAX_SCANS 11-11
  DS_TOTAL_MEMORY 11-10
  MAX_PDQ_PRIORITY 11-10
  shared-memory residency 11-15
  SQL statement cache usage 1-93, 11-11, 11-20
defined 11-1
forcing a checkpoint 11-5
freeing memory segments 11-12
killing
distributed transactions 11-23
session 11-23
marking disabled dbspace as down 11-16
options
  -a 11-4
  -BC 1 11-4
  -BC 2 11-5
  -c block 11-5
  -c unblock 11-5
  -d 11-7
  -D 11-10
  -e 1-93, 11-11
  -F 11-12
  -I 8-1
  -j 11-13
  -j -U 11-14
  -k 11-13
  -l 11-15
  -m 11-13, 11-14
  -M 11-10
  -n 11-16
  -O 11-16
  -P 11-17
  -Q 11-10
  -r 11-16
  -R 11-19
  -s 11-13, 11-14
  -S 11-11
  -U 11-13, 11-14
  -W 11-20
  -y 11-22
  -y confirm action 11-3
  -z 11-23
  -Z 11-23
PDQ 15-64
regenerating .infos file 11-19
removing virtual processors 11-17
index of options onstat utility (continued)

    -c option 15-6, 15-13
    -d option 1-28, 15-6, 15-14
    -D option 15-6, 15-18
    -d update option 15-17
    -f option 1-24, 15-6
    -F option 1-23, 15-6, 15-19
    -g ac option 15-20, 15-25
    -g afr option 15-20, 15-26
    -g all option 15-20, 15-26
    -g ath option 15-20, 15-26
    -g buf option 15-20, 15-27
    -g cac agg option 15-20
    -g cac stmt option 15-20
    -g cat option 15-20, 15-29
    -g cdr option 15-20, 15-30
    -g con option 15-20, 15-32, 15-82
    -g ddr command 15-33
    -g ddr option 15-20
    -g dic option 15-21, 15-34
    -g dis option 15-21, 15-35
    -g dll option 15-21, 15-36
    -g dmp option 15-21, 15-37
    -g dri option 15-21, 15-38
    -g dsc option 15-21, 15-38
    -g dss option 15-21
    -g dtc command 15-40
    -g dtc option 15-21
    -g env option 15-21, 15-41, 15-42
    -g ffr option 15-21, 15-43
    -g glo option 15-21, 15-44
    -g glo options A-7
    -g grp command 15-45
    -g his option 15-21, 15-50
    -g imc option 15-21
    -g ioa option 15-21, 15-52
    -g iob option 15-22, 15-53
    -g iof option 15-22, 15-54
    -g ig option 15-22, 15-55
    -g iq option 15-22, 15-55
    -g iov option 15-57
    -g iov options 15-22
    -g ipl option 15-57
    -g ipl options 15-22
    -g lap option 15-58
    -g lap options 15-22
    -g lmx option 15-22, 15-59
    -g lsc option 15-22, 15-60
    -g mem option 15-22, 15-60
    -g mgm option 11-11, 15-22, 15-61
    -g nbm option 15-22, 15-64
    -g nf command 15-65
    -g nf option 15-22
    -g nsc option 1-66, 15-22, 15-66
    -g nsd option 15-22, 15-68
    -g nss option 15-23, 15-69
    -g nta option 15-23
    -g ntd option 15-23, 15-70
    -g ntm option 15-23, 15-70
    -g ntt option 15-23, 15-71
    -g ntu option 15-23, 15-71
    -g opn option 15-23, 15-72
    -g option 15-6, 15-20
    -G option 15-6, 15-116
    -g pos option 15-23, 15-74
    -g ppf option 1-99, 15-23, 15-75
onstat utility (continued)
- `g prc option` 15-23, 15-76
- `g qst option` 15-23, 15-77
- `g que command` 15-76
- `g que option` 15-23, 15-76
- `g rbm option` 15-23, 15-78
- `g rcm option` 15-79
- `g rcm option` 15-23, 15-79
- `g rea option` 15-23, 15-82
- `g rep command` 15-82
- `g rep option` 15-23
- `g rpm command` 15-83
- `g rpm option` 15-23
- `g rsi option` 15-23, 15-85
- `g rwm option` 15-88
- `g sch option` 15-24, 15-88
- `g sds option` 15-24, 15-89
- `g seg option` 1-86, 15-24, 15-93
- `g ses option` 15-24, 15-94
- `g sele option` 15-24, 15-98
- `g smb option` 15-24, 15-98
- `g smx option` 15-24, 15-99
- `g spi option` 15-24, 15-101
- `g sql option` 15-24, 15-102
- `g src option` 15-24, 15-103
- `g scs all option` 15-24
- `g ssc option` 1-94, 15-24, 15-104
- `g ssc pool option` 1-95, 15-24
- `g stk option` 15-24, 15-106
- `g stmt option` 15-25, 15-106
- `g stq option` 15-25, 15-107
- `g stq option` 15-25, 15-107
- `g sync option` 15-25, 15-108
- `g tpf option` 15-25, 15-110
- `g ufr option` 15-25, 15-110
- `g vpcache option` 15-25, 15-112
- `g wai option` 15-25, 15-113
- `g wmx option` 15-25, 15-114
- `g wst option` 15-25, 15-114
- `-h option` 15-6, 15-117
- `-i option` 15-6, 15-119
- `-j option` 15-119
- `-k option` 1-54, 15-6, 15-121
- `-l option` 1-55, 13-2, 15-6, 15-122
- `-m option` 1-65, 15-6, 15-125, E-1
- `-o option` 15-6, 15-125
- `-O option` 1-72, 15-6, 15-126
- `-p option` 1-30, 15-127
- `-P option` 15-7, 15-130
- `-r option` 15-7, 15-131
- `-R option` 15-7, 15-134
- `-s option` 15-7, 15-136
- `-t option` 15-7, 15-138
- `-T option` 15-138
- `-u option` 15-7, 15-139
- `-x option` 11-23, 15-7, 15-142
- `-X option` 15-7, 15-144
- `-z option` 15-7, 15-146
- `defined` 15-3
- `displaying` chunk information 15-14
- `global transactions` 15-116
- `ONCONFIG file` 15-6
- `filename_dest` 15-7
- `filename_source` 15-7
- `freeing blobpages` 15-17
- `header` 15-8

onstat utility (continued)
monitoring
PDEQ 15-61
monitoring options 15-7
no options 15-8
print file info about B-tree scanner subsystem and thread 15-13
repeated execution
-r option 15-7
seconds parameter 15-8
syntax 15-4
table of options 15-5
terminating interactive mode or repeating sequence 15-119
using SMI tables for onstat information 2-30
using with shared-memory source file 15-7
ontape utility
LTAPEBLK, uses 1-58, 1-97
LTAPEDEV, uses 1-58
LTAPESIZE, uses 1-59
TAPEDEV, uses 1-98
TAPESIZE, uses 1-99
onuilmc protocol 1-69
onunload utility
LTAPEBLK, uses 1-58, 1-97
LTAPEDEV, uses 1-59
LTAPESIZE, uses 1-59
TAPEDEV, uses 1-98
TAPESIZE, uses 1-99
OPCACHEMAX configuration parameter 1-72
OpenAdmin Tool for IDS 6-2
OPTGOAL configuration parameter 1-73
OPTCOMPNOD
configuration parameter 1-73
environment variables 1-73
Optical storage and STAGEBLOB configuration parameter 1-92
Optimizing Subsystem memory cache 15-6, 15-126, 15-127
Optimizing hash and nested-loop joins 1-73
Output shared memory onstat -o 15-125

P
Page
bitmap page 4-22
blobspace blobpage 4-22
blobspace free-map page 4-22
components of dbspace page 4-12
compression 4-15
dbspase blobpage 4-21
dbspase page types 4-8, 4-9
definition of full page 4-14
free page, defined 4-9, 4-10
page types in extent 4-8, 4-9
rebalance of index page 4-20
size, shown with onstat -b 15-10
structure and storage of 4-12
Page compression 4-15
Page header, length of 4-5
PAGECONFIG reserved page 1-5, 7-12
Page-cleaner threads
codes for activity state 15-19
monitoring activity 15-6, 15-19
numbers 1-23
Parallel database queries
gate information 15-63
Parallel database queries (continued)
MGM resources 15-64
monitoring resources 15-63
monitoring resources allocated 15-64
Partition profile statistics 15-23
Partnum field in systables 4-6
Passwords, encrypted, not shown in onstat -g sql 15-102
Pathname, specifying 14-7, 14-10, 14-12
PBDELETE logical-log record 5-10
PBINSERT logical-log record 5-10
PC_HASHSIZE configuration parameter 1-75
PC_POOLSIZE configuration parameter 1-75
PDELETE record subtype (SLOB) 5-15
PDINDEX logical-log record 5-10
PDQ
CPU VPs 11-17
DS_MAX QUERIES configuration parameter 1-37
DS_MAX_SCANS configuration parameter 1-38
DS_TOTAL_MEMORY configuration parameter 1-40
information 2-17
MAX_PDQ_PRIORITY configuration parameter 1-62
PDQPRIORITY configuration parameter 1-62
PDQPRIORITY configuration parameter 1-62
Pending transaction 15-142
PER_STMT_EXEC memory duration pool 15-22
PER_STMT_PREP memory duration pool 15-22
PERASE logical-log record 5-11
PGALTER logical-log record 5-10
PGMODE logical-log record 5-11
PH ALERT table 3-3
PH GROUP table 3-3
PH_RUN table 3-3
PH TASK table 3-1
PH_THRESHOLD table 3-5
PHYSBUFF configuration parameter 1-75
PHYSDBS configuration parameter 1-76
PHYSFILE configuration parameter 1-76
Physical log
backing up
changes 13-4
files 13-4
changing
size and location 13-3
text editor 13-4
changing size 13-4
checking consistency 7-12
root dbspace 4-2
size 1-76
using non-default page size 13-4
Physical-log buffer
dbspace location 1-76
size 1-75
PLOG_OVERFLOW_PATH configuration parameter 1-77
PNGALIGN log record 5-11
PNLOCKID logical-log record 5-11
PNSIZES logical-log record 5-11
Poll threads, printing data 15-22
Pools
allocated fragments 15-20
free fragments 15-21
printing memory statistics 15-22
SQL statement cache 15-24
PREPARE logical-log record 5-11
Preventing long transactions 1-60
Primary key, use in fragmented table 4-14
Printing
active threads 15-20
Printing (continued)
AIO global information 15-22
allocated memory fragments 15-20, 15-25
block bitmap
nonresident segments 15-22
resident segments 15-23
client shared-memory status 15-22
conditions with waiters 15-20
data-distribution cache 15-21
diagnostics, onmode -1 1-1
Enterprise Replication
data sync threads 15-21
database log reader 15-20
delete table cleaner 15-21
global catalog 15-20
grouper information 15-21
network statistics 15-22
queued events 15-23
receive manager 15-23
RQM statistics 15-23
free fragments, shared memory pool 15-25
global multithreading 15-21
global transactions 15-116
HDR information 15-21
light scans 15-22
loaded dynamic libraries 15-21
locked mutexes 15-22
MaxConnect information 15-21
memory fragments 15-20
memory statistics 15-22
memory usage of SQL statements 15-25
MGM resources 15-22
mutexes with waiters 15-25
network statistics 15-23
network user times 15-23
onconfig.std file 1-5
partition profile 15-23
partitions, open 15-23
pending I/O operations 15-22
queue statistics 15-23
queue stream information 15-25
read/write mutexes 15-23
ready threads 15-23
sb space information 15-24
semaphores, spins, busy waits 15-24
session ID 15-24
shared memory
poll threads 15-22
segment statistics 15-24
session status 15-23
sleeping threads 15-24
spin locks 15-24
SPL routine cache 15-23
SQL information 15-24
SQL statement cache 15-20, 15-24
stack use per thread 15-25
tables cached in shared-memory dictionary 15-21
threads
onstat -g ath 15-20
onstat -g tpf 15-25
user-defined aggregates 15-20
user-defined types 15-21
wait statistics 15-25
waiting threads 15-25
Priority aging, of CPU virtual processors D-7
Private environment file A-5
Index X-15
Resident shared memory
   RESIDENT configuration parameter 1-79
   turning on and off residency 11-16
   RESTARTABLE_RESTORE configuration parameter 1-79
   Restarting the database server 1-11
   Results table 3-5
   Return codes on exit 15-146
   Reuse of freed index pages 4-20
   revcdr.out file E-26
   revcdr.sh script A-3, A-8, E-26
   Reversion messages
      database server E-21, E-25
      Enterprise Replication E-26, E-27
      REVERT logical-log record 5-12
      revtestcdr.out file E-26
      RINSERT logical-log record 5-12
      ROLLBACK logical-log record 5-12
      Rolling back long transactions 1-60
      ROLWORK logical-log record 5-12
   Root dbspace
      initial chunk 1-81
      mirroring 1-65
      specifying ROOTNAME configuration parameter 1-80
      structure 4-2
      using a link 1-81
   ROOTNAME configuration parameter
      defined 1-80
      use by PHYSDBS 1-76
   ROOTOFFSET configuration parameter 1-80
   ROOTPATH configuration parameter
      defined 1-81
      specifying as a link 1-65, 1-81
   ROOTSIZE configuration parameter 1-81
   ROWID
      defined 4-13
      fragmented table 4-14
      functions as forward pointer 4-13
      locking information 15-122
      stored in index pages 4-13
   Rows
      data, storage 4-14
      displaying contents with oncheck 7-13
      storage location 4-14
   RS secondary server 15-23
   RSVEXTEN logical-log record 5-12
   RTO_SERVER_RESTART
      changing dynamically 11-21
   RTO_SERVER_RESTART configuration parameter 1-82
   RTREE logical-log record 5-12
   RUPAFT logical-log record 5-12
   RUPBEF logical-log record 5-12
   RUPDATE logical-log record 5-13

S
   SBLOB logical-log record 5-13, 5-14
   Spage structure 4-25
   SBSPACENAME configuration parameter 1-83, 1-97
   sbspaces
      -g option 14-16
      adding a chunk 14-4
      changing defaults 14-16, 14-17
      cleaning up references 14-18
      creating with onspaces 14-11
      default name 1-83
      dropping a chunk 14-19
      dropping a sbpace 14-20
sbspaces (continued)
ending mirroring 14-24
maximum number 14-6, 14-8, 14-12, 14-17
metadata area
size and offset 14-12, 14-14
structure 4-24
multiple chunk 4-25
naming conventions 14-12
onstat -d usage 15-15, 15-17
onstat -g smb 15-24
reserved area 4-24
sbpage structure 4-25
starting mirroring 14-22
structure 4-23
temporary 1-84, 14-13, 15-15
creating 14-13
user-defined data statistics 1-97
SBSPACETEMP configuration parameter 1-84, 14-13
Scheduler group names
PH_GROUP table 3-3
Scheduler messages
PH_ALERT table 3-3
Scheduler task historical data
Results table 3-5
Scheduler task information
PH_RUN table 3-3
PH_TASK table 3-1
Scheduler task thresholds
PH_THRESHOLD table 3-5
Schema Editor 6-4
Screen reader
reading syntax diagrams G-1
Scripts
concdr.sh A-4
ex_alarm.sh 1-14, C-1
log_full 1-14, C-1
no_log 1-14
recvcr.sh A-8
SD secondary server 15-24
SDS_ENABLE configuration parameter 1-84
SDS_PAGING configuration parameter 1-84
SDS_TEMPDBS configuration parameter 1-85
SDS_TIMEOUT configuration parameter 1-85
Searching for patterns 15-24
security
local 1-86
SECURITY_LOCALCONNECTION configuration parameter 1-86
Semaphores 15-24
Server Studio 6-4
SERVERNUM configuration parameter 1-86
Session information
global transactions 15-144
setting environment variables 1-12
SMI tables 2-20, 2-21
using onstat -g ses 15-24
using onstat -g sql 15-24
SET EXPLAIN statement
setting dynamically 11-22
SET STATEMENT CACHE statement 1-93, 11-11
Setting data replication type 11-7
Shared library files  A-5
Shared memory
adding segment with onmode 11-4
base address 1-87
buffer, frequency of flushing 1-20
buffer, maximum number 1-21
Shared memory (continued)
changing
decision-support parameters 11-10
residency with onmode 11-15
dumps 1-41, 1-42
examining with SMI 2-3
freed fragments 15-21
initializing 9-1
monitoring 15-3
network poll threads 15-22
network status 15-23
physical-log buffer 1-75
pool statistics 15-22
resident portion, flag 1-79
saving copy of with onstat 15-6
segment statistics 15-24
segments, dynamically added, size 1-87
SERVERNUM configuration parameter 1-86
size displayed by onstat 15-8
status 15-22
virtual segment, initial size 1-89
Shared-memory dictionary 15-21
SHMADD configuration parameter
64-bit addressing 1-86
defined 1-86
SHMBASE configuration parameter 1-87
shm file
DUMPSHMEM configuration parameter 1-42
shm.stderr A-8
SHMTOTAL configuration parameter 1-88
SMHVIRT_ALLOCSEG configuration parameter 1-88
SHMVIRTSIZE configuration parameter 1-89
shortcut keys
keyboard G-1
Shutting down the database server 11-13, 11-14
SINGLE_CPU_VP configuration parameter 1-90
Size
chunk 14-4, 14-5
index fragments 4-8
metadata 14-12
offset 14-7, 14-10, 14-12, 14-17
Sleeping threads 15-24
sm_versions.std file A-8
Smart large objects
buffer pool 1-21
cleaning up references 14-18
default name 1-83
logging 14-15
logical-log records 5-14
user-defined data statistics 1-97
SMX connections 15-24
snmpd.log file A-9
Specify
modified pages, percentage
LRU queue 13-5
Specifying pathname 14-4
Spin locks, printing 15-24
SPL routine cache 15-23
SQL Editor 6-4
SQL profile
information 2-25
SQL statement cache
enabling the cache 1-93, 11-11
flushing the cache 11-11
inserting
key-only entries 1-94, 11-20
qualified statements 1-94, 11-20
SQL statement cache (continued)
  limiting the cache size 1-94
  memory pools 1-95
  printing contents 15-20, 15-24
  specifying number of hits 1-93, 11-20
  turning off the cache 11-11
  turning on the cache 1-93, 11-11

SQL statements
  printing information 15-24
  printing memory usage 15-25
  SET STATEMENT CACHE 1-93, 11-11
  UPDATE STATISTICS 1-97

SQLCA, warning flag when fragment skipped during query 1-24
sqlhosts file or registry
  defined A-9
  multiple dbservernames 1-26
sqlmux, multiplexed connections in NETTYPE configuration parameter 1-69

SQLTRACE
  onstat -g his option 15-21
SQLTRACE configuration parameter 1-91
  Stack use 15-25
STACKSIZE configuration parameter 1-92
STAGEBLOB configuration parameter 1-92
  Starting database server with oninit 9-1
  Starting the database server online 11-13, 11-14
Statistics
  VP memory block cache 15-25
  STMT_CACHE configuration parameter 1-93
  STMT_CACHE environment variable 11-11
  STMT_CACHE environment variables 1-93
  STMT_CACHE_HITS configuration parameter 1-93, 11-20
  STMT_CACHE_NOLIMIT configuration parameter 1-94
  STMT_CACHE_NUMPOOL configuration parameter 1-95
  STMT_CACHE_SIZE configuration parameter 1-95
Storage manager
  xbsa.messages log A-9
Symbolic links
  using with shared libraries A-5
  using with TAPEDEV configuration parameter 1-98
SYNC logical-log record 5-13
Syntax diagrams
  reading in a screen reader G-1
sysadmind database
  Results table 3-5
  tables
    command_history 3-5
    PH_ALERT 3-3
    PH_GROUP 3-3
    PH_RUN 3-3
    PH_TASK 3-1
    PH_THRESHOLD 3-5
sysadminfo table 2-6
SYSSALARMPROGRAM configuration parameter 1-95
sysaudit table 2-7
syscheckpoint table 2-8
syschkio table 2-7
syschunks table 2-8
sysconfig table 2-9
sysdatabases table 2-10
sysdbslocalsec table 2-10
sysdbspaces table 2-10
sysdri table 2-11
sysdual table 2-12
sysenv table 2-12
sysenvses table 2-12
sysessions table 2-22
sysextents table 2-12
sysextspaces table 2-13
sysha_latency table 2-13
sysha_type table 2-14
sysha_workload table 2-14
sysipl table 2-15
syslocks table 2-15
syslogs table 2-16
sysmaster database
  buildsmi.xxx file A-4
  defined 2-2
  failure to build A-4
  functionality of 2-2
  initialization 9-1
  list of topics covered by 2-3
  SMI tables 2-3
  space required to build 2-2
  sysexstspaces 2-13
  types of tables 2-2
  warning 2-2
  when created 2-2
sysmgminfo table 2-17
sysnetclienttype table 2-17
sysnetglobal table 2-17
sysnetworkio table 2-18
sysonlinelog table 2-19
sysprofile table 2-19
sysprof table 2-20
sysssqllog table 2-21
SYSSSBSPACENAME configuration parameter 1-97
sysccblst table 2-21
sysesprof table 2-21
sysmxm table 2-23
sysmxmsses table 2-24
sysssqltrace table 2-24
sysssqltrace_info table 2-25
sysssqltrace_iter table 2-26
syssscssst table 2-26
sysssrcds table 2-26
systrabnames table 2-27
System catalog tables
  disk space allocation 4-26
  listing 7-6
  oncheck -cc 7-8
  oncheck -pc 7-8
  sydsdistrib 1-97
  sysfragments table 4-6
  systraceclasses B-1
  systracemsgs B-1
  tracking 4-27
  tracking a new database 4-27
  tracking a new table 4-28
System page size, specifying 1-22
System-monitoring interface
  accessing SMI tables 2-4
  defined 2-2
  list of SMI tables 2-4
  locking 2-4
  obtaining onstat information 2-30
SPL 2-4
  tables
    defined 2-3
    list of supported 2-4
    sysadminfo 2-6
    sysaudit 2-7
    syscheckpoint 2-8
System-monitoring interface (continued)

Table (continued)

temporary (continued)
  message reporting cleanup  E-7
  TABLOCKS logical-log record 5-13
tail -t command  E-1
Tape device, block size 1-58
TAPEBLK configuration parameter 1-97
TAPEDEV configuration parameter
  defined 1-98
  using a symbolic link 1-98
TAPESIZE configuration parameter 1-99

Tbspace
  displaying (onstat -t or -T) 15-7, 15-138
  monitoring
    bspace statistics  1-99
    with SMI  2-20
  number 4-6, 15-139
  table fragment 4-6, 4-15

TBLSPACE configuration parameter 15-23

Tbspace number
  defined 4-6
  elements 4-7
  includes dbspace number 4-6
table fragment 4-6

Tbspace tblspace
  bitmap page  4-7
  location in a chunk 4-3
  location in root dbspace 4-2
  tracking new tables 4-27

TBLSPACE_STATS configuration parameter 1-99

TBLTBLFIRST configuration parameter 1-100
TBLTBLNEXT configuration parameter 1-100

Template
  ac_config.std file  A-3
  onconfig.std file  A-7

Temporary dbspace
  creating with onspaces 14-10
  DBSPACETEMP configuration parameter 14-10

Temporary dbspace
  creating with onspaces 14-13
  onstat -d 15-15
  DBSPACETEMP configuration parameter 1-84

temporary dbspaces
  creating with onspaces 14-11

Temporary smart large object
  default sbspace 1-84
  DBSPACETEMP configuration parameter 1-84

Temporary tables
  DBSPACETEMP configuration parameter 1-28
  extent size doubling 4-11
  oninit utility 9-2
  rules for use 1-28

TEMPTAB_NOLOG configuration parameter 1-100

TEXT and BYTE data
  blob descriptor 4-13, 4-22
  modifying storage 4-22
  page descriptor 4-22
  size limitations 4-22
  storage on disk 4-21, 4-22
  updating 4-22
  when modified 4-22
  when written 4-22

Threads
  data sync 15-21
dump stack 15-24
  onstat -g tpf 15-25
  onstat -X usage 15-7, 15-144
Threads *(continued)*
- poll 15-22
- printing sqlmain 15-20
- ready 15-23
- sleeping 15-24
- waiting 15-25

Tightly-coupled mode 15-144

Time stamp
- Blobspace blopage 4-23
- defined 4-26

Time-out condition 15-19

Trace class B-1

Trace message B-1

Tracepoints B-1

Transaction manager
- loosely-coupled mode 15-144
- tightly-coupled mode 15-144

Transaction Replicate Group E-26

Transactions
- heterogeneous commit 1-52
- kill with onmode -Z 11-23
- pending 15-142
- XID 15-144

Trapping errors with onmode B-1

TRUNCATE 5-10

TRUNCATE log record 5-10

Truncate table messages E-29

Turning
- large number of users 1-68
- use of NETTYPE configuration parameter 1-67

Turning on SQL statement cache 11-11

Two-phase commit protocol, killing distributed transactions 11-23

TXTIMEOUT configuration parameter
- defined 1-101
- onmode utility 11-23

U

UDINSERT
- logical-log record 5-13
- record subtype 5-15

UDUPAFT
- logical-log record 5-14
- record subtype 5-15

UDUPBEF
- logical-log record 5-14
- record subtype 5-16

UDWRITE
- logical-log record 5-14
- record subtype 5-16

Unblocking database server 11-5

Unbuffered disk space
- UNIX 14-4
- Windows 14-3, 14-7, 14-8

UNDO logical-log record 5-14

UNDOBLDC log record 5-14

UNIQID logical-log record 5-14

UNIX
- buffered disk space 14-4
- interrupt signal 6-1
- unbuffered disk space 14-4
- using onspaces 14-2

UNSECURE_ONSTAT configuration parameter 1-101

UPDATE STATISTICS statement 1-37, 1-39, 1-97

Updating blobspace statistics 15-17

UPDBEF logical-log record 5-14

USELASTCOMMITTED
- changing dynamically 11-21

USELASTCOMMITTED configuration parameter 1-101

USEOSTIME parameter 1-102

User connections, monitoring E-11

User session
- monitoring with SMI 2-22
- status codes 15-140

User-defined aggregates, printing definitions 15-20

User-defined routines, debugging B-1

User-defined type
- data distributions 1-97
- printing ER information 15-21

Utilities
- -V option 6-2
- -version option 6-2

See also Command-line utilities
- changing parameter values 1-11
- gcore 1-41, 1-42
- IBM Informix Server Administrator 6-3
- ON-Monitor utility 1-12, 1-22, 12-1, 12-4

oncheck 7-1, 7-20

ondblog utility 8-1

oninit 9-1

onlog 10-1, 10-4

onmode 11-1, 11-22

onmode and PDQ 15-64

onparams 13-1, 13-6

onspaces 14-1, 14-22

onstat -g options 15-20

onstat utility 15-3, 15-146

- g env option 15-41, 15-42

- g mgm option 15-61

- g seg option 1-86

option 6-2

quick reference 6-1

V

VARCHAR data type
- 4-bit bit map requirement 4-8, 4-9
- implications for data row storage 4-14
- indexing considerations 4-20
- storage considerations 4-13

Verifying backups 1-12

Violations table
- messages E-3, E-19

Virtual processors
- adding or removing with onmode 11-17
- AIO statistics 15-22
- limits 11-17
- number in
  - AIO class D-7
  - CPU class D-8
  - priority aging D-7
  - processor affinity 1-66

Visual disabilities
- reading syntax diagrams G-1

VP memory block cache statistics 15-25

VP_MEMORY_CACHE_KB
- changing dynamically 11-21

VP_MEMORY_CACHE_KB configuration parameter
- defined 1-103

VPservername.nnx file A-9

VPCLASS configuration parameter 1-54

AFF_NPROCS D-2
VPCLASS configuration parameter (continued)

- AFF_SPROC D-2
- default values 1-104
- defined 1-104
- in ONCONFIG file 1-104
- NOAGE D-7
- NUMAIOVPS D-7
- onmode utility 11-18
- reserved names 1-105
- setting maximum VPs 1-107
- setting number of VPs 1-107
- setting processor affinity 1-107
- user-defined classes 1-108

W

- Waiters, using onstat -g con 15-20
- Waits
  - threads 15-25
  - virtual processor 15-24
- Warnings
  - buildsmi script 2-3
  - when fragment skipped during query processing 1-24
- White space in ONCONFIG file 1-4
- Windows
  - adding or removing virtual processors 11-18
  - buffered disk space 14-4
  - unbuffered disk space 14-3, 14-7, 14-8
  - using onspaces 14-2
- WSTATS configuration parameter 1-108

X

- XAPREPAIR logical-log record 5-14
- xbsa.messages log A-9