INFORMIX®-Universal Server

Archive and Backup Guide

Version 9.1
March 1997
Part No. 000-4807

Published by INFORMIX® Press

Informix Software, Inc.
4100 Bohannon Drive
Menlo Park, CA 94025

Documentation Team: Bob Berry, Twila Booth, Evelyn Eldridge-Diaz

Table of Contents

Section II The ON-Archive Recovery System

Chapter 2 What Is ON-Archive?

Chapter 3 Configuring ON-Archive

Chapter 5 Creating an Archive

Chapter 6 Backing Up the Logical Log

Introduction

Read this introduction for an overview of the information provided in this manual and for an understanding of the documentation conventions used.

Important: *The current archive and backup terminology used within this manual is not compatible with X/Open terminology. However, other related INFORMIX-Universal Server documents comply with X/Open standard terminology.*

About This Manual

The *INFORMIX-Universal Server Archive and Backup Guide* describes the components, features, and tasks involved in designing and using a recovery system for a full-scale production system. This manual describes the tools that are provided with INFORMIX-Universal Server to assist you in archiving and recovering data, specifically the ON-Archive system and the **ontape** utility.

Organization of This Manual

This manual includes the following chapters:

- This Introduction provides an overview of the manual and describes the documentation conventions used.
- Chapter 1, "What Is a Universal Server Recovery System?" explains what an archive is, what a logical-log back up is, and what a restore is. It also compares the two Universal Server recovery systems: the ON-Archive system and the **ontape** utility.
- Chapter 2, "What Is ON-Archive?" describes the components, basic concepts, and features of the ON-Archive system.

- Chapter 3, "Configuring ON-Archive," describes the syntax and purpose of each of the ON-Archive configuration parameters. It describes how to change them, if you need to, and tells you where the configuration files are located.
- Chapter 4, "Using ON-Archive," tells you how to start and stop the ON-Archive programs, how to interact with the system to execute commands, and how to prepare media for Universal Server archives and logical-log backups.
- Chapter 5, "Creating an Archive," tells you how to prepare for and create an archive of Universal Server data using the ON-Archive system. It describes how to use the ARCHIVE command.
- Chapter 6, "Backing Up the Logical Log," tells you how to prepare for and back up the Universal Server logical-log files using ON-Archive. It describes how to use the BACKUP command.
- Chapter 7, "Restoring Universal Data," tells you how to recover Universal Server data from archive and backup media. It describes how to use the RETRIEVE/DBSPACESET and RETRIEVE/LOGFILES commands.
- Chapter 8, "Administering ON-Archive,"describes the ON-Archive commands that enable you to administer ON-Archive media and access and modify information in the ON-Archive catalog.
- Chapter 9, "ON-Archive Utilities and Scripts," describes how to use the ON-Archive utility programs and command scripts.
- Chapter 10, "The ON-Archive Catalog," describes the ON-Archive catalog tables, which are part of the Universal Server **sysmaster** database.
- Chapter 11, "ON-Archive Error Messages," lists the error messages that can occur while using ON-Archive. For each message, it describes the probable cause of the error and what you can do to correct the error condition.
- Chapter 12, "Configuring ontape," describes how to set Universal Server configuration parameters to use the **ontape** utility as your Universal Server recovery system.
- Chapter 13, "Using ontape," describes how to use the **ontape** utility to archive data, back up logical-log files, and recover data from backup media. It includes the syntax and usage of the **ontape** command.

Types of Users

This manual is written for system administrators, network administrators, and database administrators who work with Universal Server.

Software Dependencies

This manual assumes that you are using INFORMIX-Universal Server, Version 9.1, as your database server.

In this manual, all instances of Universal Server refer to INFORMIX-Universal Server.

Assumptions About Your Locale

Informix products can support many languages, cultures, and code sets. All culture-specific information is brought together in a single environment, called a GLS (Global Language Support) locale.

This manual assumes that you are using the default locale, **en_us.8859-1**. This locale supports U.S. English format conventions for dates, times, 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 ñ.

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(s). For instructions on how to specify a nondefault locale, additional syntax, and other considerations related to GLS locales, see the *Guide to GLS Functionality*.

Demonstration Database

The DB-Access utility, which is provided with your Informix database server products, includes a demonstration database called **stores7** that contains information about a fictitious wholesale sporting-goods distributor. Sample command files are also included.

Many examples in Informix manuals are based on the **stores7** demonstration database. The **stores7** database is described in detail and its contents are listed in Appendix A of the *Informix Guide to SQL: Reference*.

The script that you use to install the demonstration database is called **dbaccessdemo7** and is located in the **$INFORMIXDIR/bin** directory. For a complete explanation of how to create and populate the demonstration database on your database server, refer to the *DB-Access User Manual*.

Major Features

The Introduction to each Version 9.1 product manual contains a list of major features for that product. The Introduction to each manual in the Version 9.1 *Informix Guide to SQL* series contains a list of new SQL features.

Major features for Version 9.1 Informix products also appear in release notes.

Documentation Conventions

This section describes the conventions that this manual uses. These conventions make it easier to gather information from this and other Informix manuals.

The following conventions are covered:

- Typographical conventions
- Icon conventions
- Command-line conventions

Typographical Conventions

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

Convention	Meaning
KEYWORD	All keywords appear in uppercase letters in a serif font.
italics	Within text, new terms and emphasized words appear in italics. Within syntax diagrams, values that you are to specify appear in italics.
boldface	Identifiers (names of classes, objects, constants, events, functions, program variables, forms, labels, and reports), environment variables, database names, filenames, table names, column names, icons, menu items, command names, and other similar terms appear in boldface.
`monospace`	Information that the product displays and information that you enter appear in a monospace typeface.
KEYSTROKE	Keys that you are to press appear in uppercase letters in a sans serif font.
♦	This symbol indicates the end of feature-, product-, platform-, or compliance-specific information.
	This symbol indicates a unique identifier (primary key) for each table.

Tip: *When you are instructed to "enter" characters or to "execute" a command, immediately press* RETURN *after the entry. When you are instructed to "type" the text or to "press" other keys, no* RETURN *is required.*

Icon Conventions

Throughout the documentation, you will find text that is identified by several different types of icons. This section describes these icons.

Comment Icons

Comment icons identify warnings, important notes, or tips. This information is always displayed in italics.

Icon	Description
	The *warning* icon identifies vital instructions, cautions, or critical information.
	The *important* icon identifies significant information about the feature or operation that is being described.
	The *tip* icon identifies additional details or shortcuts for the functionality that is being described.

Command-Line Conventions

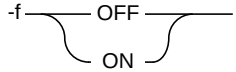
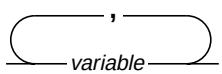
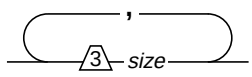
This section defines and illustrates the format of commands that are available in Informix products. These commands have their own conventions, which might include alternative forms of a command, required and optional parts of the command, and so forth.

Each diagram displays the sequences of required and optional elements that are valid in a command. A diagram begins at the upper-left corner with a command. It ends at the upper-right corner with a vertical line. Between these points, you can trace any path that does not stop or back up. Each path describes a valid form of the command. You must supply a value for words that are in italics.

You might encounter one or more of the following elements on a command-line path.

Element	Description
command	This required element is usually the product name or other short word that invokes the product or calls the compiler or preprocessor script for a compiled Informix product. It might appear alone or precede one or more options. You must spell a command exactly as shown and use lowercase letters.
variable	A word in italics represents a value that you must supply, such as a database, file, or program name. A table following the diagram explains the value.
-flag	A flag is usually an abbreviation for a function, menu, or option name or for a compiler or preprocessor argument. You must enter a flag exactly as shown, including the preceding hyphen.
.ext	A filename extension, such as **.sql** or **.cob**, might follow a variable that represents a filename. Type this extension exactly as shown, immediately after the name of the file. The extension might be optional in certain products.
(. , ; + * - /)	Punctuation and mathematical notations are literal symbols that you must enter exactly as shown.
' '	Single quotes are literal symbols that you must enter as shown.
Privileges p. 5-17 Privileges	A reference in a box represents a subdiagram. Imagine that the subdiagram is spliced into the main diagram at this point. When a page number is not specified, the subdiagram appears on the same page.
ALL	A shaded option is the default action.
	Syntax within a pair of arrows indicates a subdiagram.
	The vertical line terminates the command.

(1 of 2)

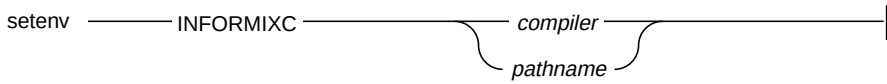
Element	Description
	A branch below the main path indicates an optional path. (Any term on the main path is required, unless a branch can circumvent it.)
	A loop indicates a path that you can repeat. Punctuation along the top of the loop indicates the separator symbol for list items.
	A gate (3) on a path indicates that you can only use that path the indicated number of times, even if it is part of a larger loop. Here you can specify *size* no more than three times within this statement segment.

(2 of 2)

How to Read a Command-Line Diagram

Figure 1 shows a command-line diagram that uses some of the elements that are listed in the previous table.

Figure 1
Example of a Command-Line Diagram



To construct a command correctly, start at the top left with the command. Then follow the diagram to the right, including the elements that you want. The elements in the diagram are case sensitive.

Figure 1 diagrams the following steps:

1. Type the word `setenv`.
2. Type the word `INFORMIXC`.
3. Supply either a compiler name or pathname.

   After you choose *compiler* or *pathname*, you come to the terminator. Your command is complete.
4. Press RETURN to execute the command.

Additional Documentation

For additional information, you might want to refer to the following types of documentation:

- On-line manuals
- Printed manuals
- Error message files
- Documentation notes, release notes, and machine notes

On-Line Manuals

A CD that contains Informix manuals in electronic format is provided with your Informix products. You can install the documentation or access it directly from the CD. For information about how to install, read, and print on-line manuals, see either the installation guide for your product or the installation insert that accompanies the documentation CD.

The documentation set that is provided on the CD describes Universal Server, its implementation of SQL, and its associated application-programming interfaces. For an overview of the manuals in the Universal Server documentation set, see *Getting Started with INFORMIX-Universal Server*.

Printed Manuals

The Universal Server documentation set describes Universal Server, its implementation of SQL, and its associated application-programming interfaces. For an overview of the manuals in the Universal Server documentation set, see *Getting Started with INFORMIX-Universal Server*.

To order printed manuals, call 1-800-331-1763 or send email to moreinfo@informix.com.

Please provide the following information:

- The documentation that you need
- The quantity that you need
- Your name, address, and telephone number

Error Message Files

Informix software products provide ASCII files that contain all the Informix error messages and their corrective actions. To read the error messages in the ASCII file, Informix provides scripts that let you display error messages on the screen (**finderr**) or print formatted error messages (**rofferr**). For a detailed description of these scripts, see the Introduction to the *Informix Error Messages* manual.

Documentation Notes, Release Notes, Machine Notes

In addition to printed documentation, the following on-line files, located in the **$INFORMIXDIR/release/en_us/0333** directory, supplement the information in this manual.

On-Line File	Purpose
ARCHIVEDOC_9.1	The documentation-notes file describes features that are not covered in this manual or that have been modified since publication.
SERVERS_9.1	The release-notes file describes feature differences from earlier versions of Informix products and how these differences might affect current products. This file also contains information about any known problems and their workarounds.
IUNIVERSAL_9.1	The machine-notes file describes any special actions that are required to configure and use Informix products on your computer. Machine notes are named for the product described.

Please examine these files because they contain vital information about application and performance issues.

Compliance with Industry Standards

The American National Standards Institute (ANSI) has established a set of industry standards for SQL. 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, on INFORMIX-Universal Server. In addition, many features of Universal Server comply with the SQL-92 Intermediate and Full Level and X/Open SQL CAE (common applications environment) standards.

Informix Welcomes Your Comments

Please tell us what you like or dislike about our manuals. To help us with future versions of our manuals, we want to know about corrections or clarifications that you would find useful. Include the following information:

- The name and version of the manual that you are using
- Any comments that you have about the manual
- Your name, address, and phone number

Write to us at the following address:

Informix Software, Inc.
SCT Technical Publications Department
4100 Bohannon Drive
Menlo Park, CA 94025

If you prefer to send email, our address is:

doc@informix.com

Or send a facsimile to the Informix Technical Publications Department at:

415-926-6571

We appreciate your feedback.

Section IV

What Is a Universal Recovery System?

Chapter

1

What Is a Universal Server Recovery System?

This chapter explains the central concepts of a data recovery system for the INFORMIX-Universal Server. It describes the basic tasks of using an Universal Server recovery system by answering the following questions:

- What is an Universal Server recovery system?
- What is an archive?
- What is a logical-log backup?
- What is a restore?

Informix provides two recovery systems for Universal Server, the ON-Archive system and the **ontape** utility. The next section of this book, Chapter 2 through Chapter 11, describes the ON-Archive system. The third section, Chapter 12 and Chapter 13, describe the **ontape** utility. This chapter describes the general concepts of an Universal Server recovery system, that apply to both systems.

A companion volume to this book, the *ON-Archive Quick Start Guide*, provides a step-by-step, hands-on introduction to the ON-Archive recovery system. It guides you step-by-step through the operations of archiving and restoring data using the **stores7** demonstration database.

What Is a Universal Recovery System?

An Universal Server *recovery system* enables you to back up your Universal Server data and subsequently restore it in the event that your current data becomes corrupt or inaccessible. The causes of data corruption or loss can range from a program error to a disk crash to a disaster that damages the entire facility. A recovery system enables you to recover data that you already lost due to such mishaps.

What Is an Archive?

An *archive* is a copy of either all or some portion of the data that Universal Server manages. More precisely, an archive is a copy of one or more Universal Server *dbspaces* (*d*ata*b*ase *spaces*) and any supporting data that you might need to restore them. See the *INFORMIX-Universal Server Administrator's Guide* for a description of an Universal Server dbspace.

You create an archive of Universal Server data on tape or disk that, ideally, you store in a safe location that is separate from your computer facility. Figure 1-1 illustrates the basic concept of an Universal Server archive.

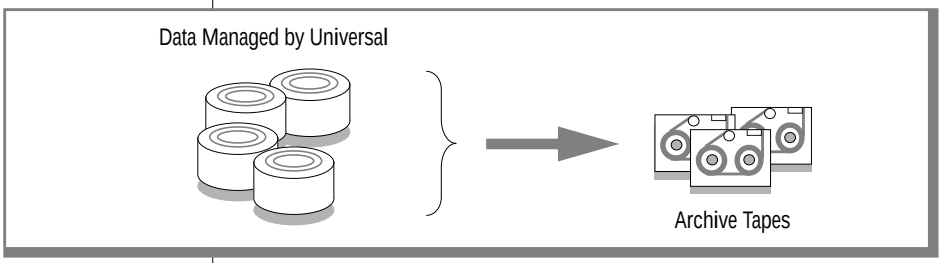


Figure 1-1
An Archive of Universal Data

Chapter 5, "Creating an Archive," explains in detail how to archive Universal Server data using ON-Archive. "Creating an Archive" on page 13-8 explains how to archive Universal Server data using the **ontape** utility.

What Is a Logical-Log Backup?

A *logical-log backup* is a copy to tape or disk of logical-log files that have become full and eligible for backup. The logical-log files store a record of Universal Server activity that occurs *between archives*.

To illustrate, when you create an archive of all your Universal Server data at time **t0** and a failure occurs later at time **t1**, you could lose any transactions that occurred between those points in time. When you specify transaction logging for your databases, however, the transactions that occur between **t0** and **t1** are stored in the logical log. Figure 1-2 on page 1-5 illustrates the function of the logical log.

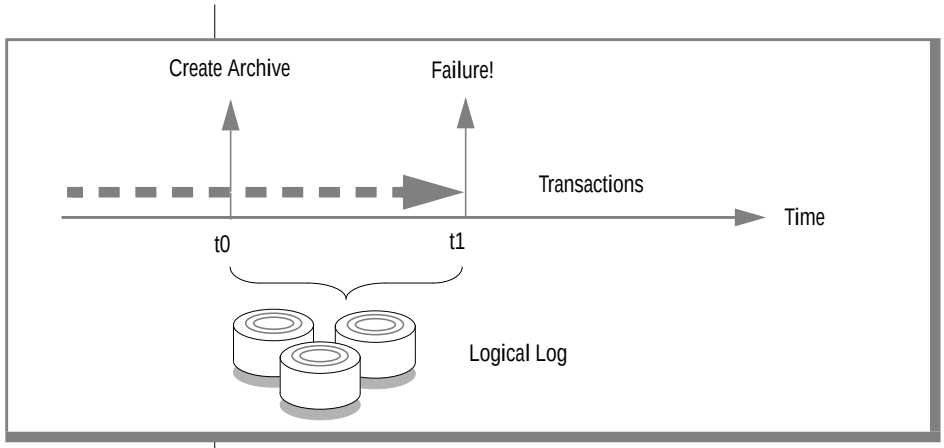


Figure 1-2
The Logical Log

Universal Server reuses the logical-log files to minimize the amount of disk space it needs for logging. Therefore, you must back up the logical-log files when they become full to allow Universal Server to free them to make room for subsequent transactions. As with archives, you can back up the logical-log files to either tape or disk. See the *INFORMIX-Universal Server Administrator's Guide* for a complete description of the logical log. Figure 1-3 illustrates a logical-log backup.

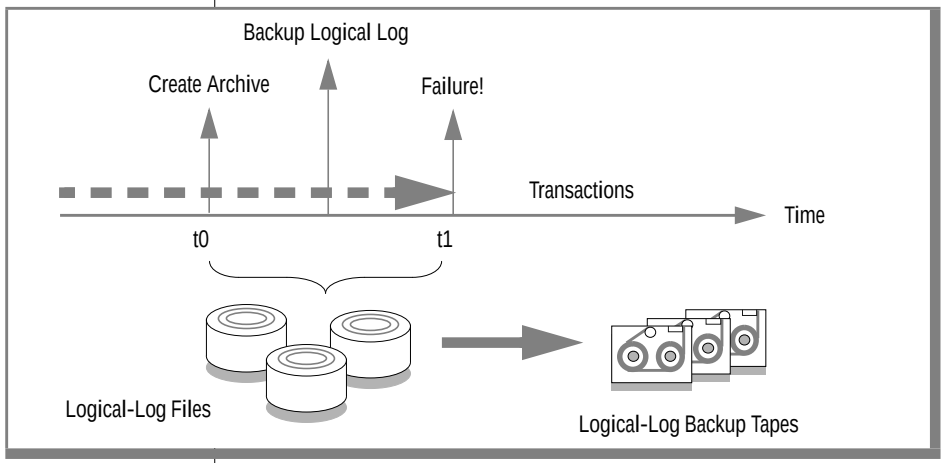


Figure 1-3
Logical-Log Backups

When you need to restore dbspaces from your archives, in most cases you must also restore all transactions from the logical-log files that you backed up after the archive. When the failure causes Universal Server to come off-line, you can usually *salvage* any logical-log files that were not backed up at the time of the failure and then restore them as well. This enables you to recover all of your Universal Server data up to the last complete transaction at the time of the failure. Chapter 6, "Backing Up the Logical Log," explains how to back up the logical log using ON-Archive. "Backing Up Logical-Log Files" on page 13-29 describes how to back up the logical log using the **ontape** utility.

What Is a Universal Server Restore?

An Universal Server *restore* re-creates Universal Server data—particularly Universal Server dbspaces—from an archive and backed-up logical-log files.

Physical and Logical Restores

You must restore Universal Server data in two operations. The first operation is a *physical* restore and the second, that must follow the first, is a *logical* restore. These two operations are defined in the following list:

- Physical restore

  A physical restore is the process of restoring dbspace or blobspace data from an archive.

- Logical restore

  A logical restore accesses a logical-log backup to re-create in the restored dbspaces any transactions that were generated after the archive. When no databases use transaction logging, or when you specified **/dev/null** as the backup device, or when you did not back up the logical-log files, you cannot restore any changes that you made to your databases after the archive.

Figure 1-4 illustrates a physical restore.

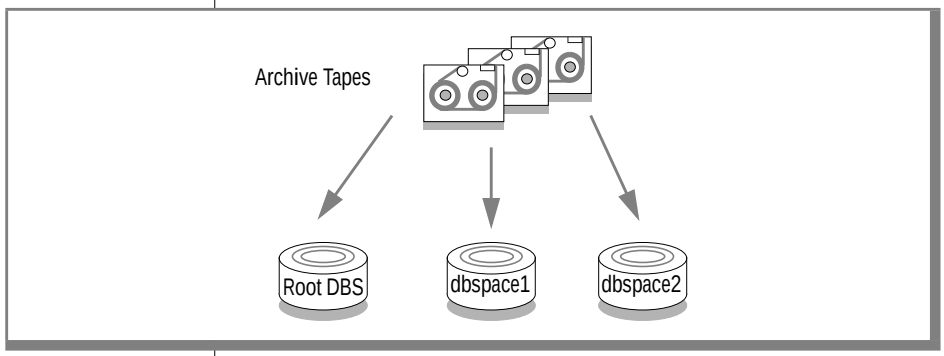


Figure 1-4
Physical Restore

You can perform a physical restore on either all Universal Server dbspaces and blobspaces or on selected dbspaces and blobspaces. For example, when you suffer a disk crash, you can restore to a new disk only those dbspaces with chunks that resided on the failed disk.

Chapter 7, "Restoring Universal Data," describes how to perform a restore using ON-Archive. "Restoring Universal Data" on page 13-39 explains how to restore Universal Server data using the **ontape** utility.

A logical restore, illustrated in Figure 1-5, restores Universal Server transactions from backed-up logical-log files for the dbspaces and blobspaces that were restored by the physical restore. Even when the dbspaces are physically restored from different archives, the logical restore rolls forward all the logical-log records for the dbspaces and blobspaces following the times of their respective archives.

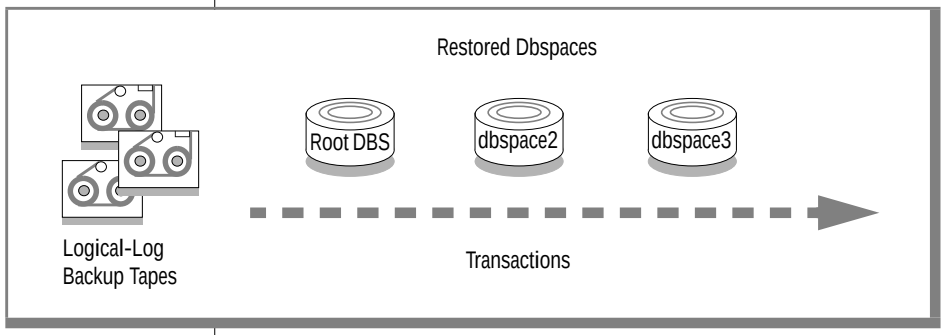


Figure 1-5
Logical Restore

What Are the Universal Server Recovery Systems?

Universal Server provides the following two recovery systems:

- ON-Archive
- The **ontape** utility

Both of these tools enable you to perform the following operations:

- Archive Universal Server data
- Back up Universal Server logical-log files
- Perform continuous backups of the logical-log files
- Restore Universal Server data from an archive
- Change the logging status of a database

ON-Archive provides the following additional features:

- Scheduling and tracking of archives and backups
- Tape reliability and storage options
- Multiple ways of securing data and access to ON-Archive
- Use of multiple tape drives simultaneously for archives and restores
- Unattended operations
- Disaster provision

Chapters 2 through 11 fully describe these additional ON-Archive features. Chapter 12, "Configuring ontape," and Chapter 13, "Using ontape," describe the **ontape** utility.

Warning: *The archive tapes produced by* ***ontape*** *and ON-*Archive *are not compatible! You cannot create an archive with* ***ontape*** *and restore it with ON-*Archive*. Also, you must not try to restore some logical logs created by one tool and then restore other logical logs from a tape created by the other tool.*

What Are the Differences Between ON-Archive and ontape?

Figure 1-6 compares ON-Archive and **ontape**.

Figure 1-6
Differences Between ON-Archive and ontape

Can it…	ON-Archive	ontape
Archive all data managed by an Universal Server database server?	yes	yes
Archive selected dbspaces?	yes	no
Restore all data managed by an Universal Server database server?	yes	yes
Restore selected dbspaces?	yes	yes
Change logging mode for databases?	yes	yes
Perform continuous logical-log file backups?	yes	yes
Archive while Universal Server is in on-line mode	yes	yes
Archive while Universal Server is in quiescent mode	yes	yes
Archive different dbspaces in parallel?	yes	no
Restore different dbspaces in parallel?	yes	no
Compress archived and backed-up data?	yes	no
Encrypt archived and backed-up data?	yes	no
Allow SQL access to archive-tracking information?	yes	no
Allow automation of archive schedule?	yes	no
Allow unattended operations?	yes	no
Automatically make copies of archives?	yes	no
Control access to storage and operations?	yes	no
Designate administrators and operator accounts?	yes	no
Prompt for specific tapes during a restore?	yes	no

When you use **ontape**, you perform the physical restore and the logical restore in a single operation. When you use ON-Archive, however, you use different commands to start a physical restore and a logical restore. This feature enables you to perform multiple physical restores—that is, restore multiple dbspaces—concurrently, assuming you have multiple devices available for the media containing the archived data. You could then follow the physical-restore operations with a single logical restore.

Although ON-Archive and **ontape** are substantially different, they are nonetheless both archive-and-recovery tools. The general description of a recovery system in this chapter applies to both ON-Archive and **ontape**, except where it is noted otherwise.

Tools You Cannot Use for Recovery

You cannot use data-migration utilities (such as **onunload** or **dbexport**) as a substitute for an Universal Server archive. None of the data-migration utilities are coordinated with the information stored in the logical-log files and, unlike archives, they do not save a copy of system-overhead information important to Universal Server.

The ON-Archive Recovery System

Section V

Chapter

2

What Is ON-Archive?

This chapter describes the components and features of the ON-Archive recovery system for the INFORMIX-Universal Server. It describes the ON-Archive programs, the Universal Server data that can be archived and backed up, and how ON-Archive organizes data and media so that you readily can retrieve any backup data you need to restore. This chapter also describes the ON-Archive storage features and the ON-Archive security features.

Overview of ON-Archive

As shown in Figure 2-1 on page 2-4, the ON-Archive recovery system has the following main components:

- Universal Server data to be archived, backed up, or restored
- ON-Archive commands and requests
- ON-Archive programs
- ON-Archive catalog
- Devices to write to and read from
- Media to store the archive or backup data

Each of these components is described in detail in this chapter.

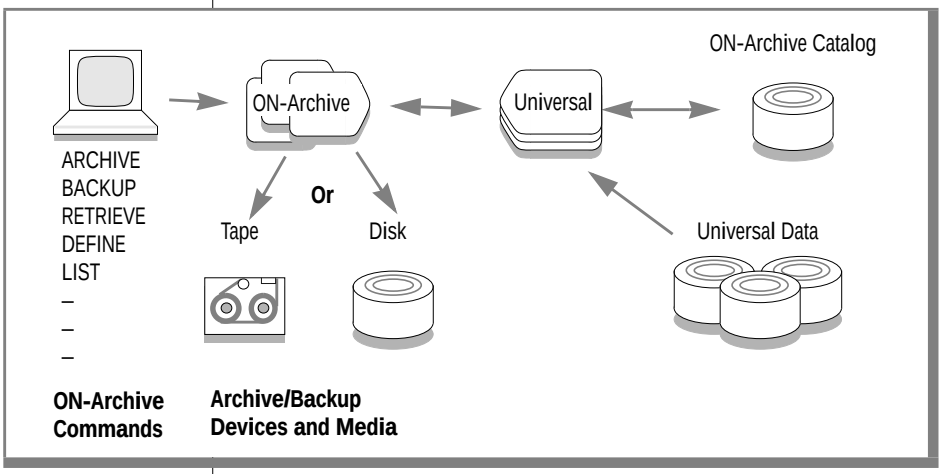


Figure 2-1
ON-Archive Components

ON-Archive operates in the following three modes:

- Attended operations

  In attended operations, ON-Archive assumes that a human operator interacts with ON-Archive and Universal Server operates in on-line mode. The operator instructs ON-Archive to perform archives, logical-log backups, and so on. ON-Archive prompts the operator to mount media as needed. When errors occur, ON-Archive sends messages to the terminal for the operator to see and act on.

- Unattended operations

  In unattended operations, an ON-Archive program, **onautovop**, acts as a *virtual operator* to perform archives and backups. The virtual operator assumes that no operator exists to mount or change media, so it does not issue a prompt for media when needed. Instead of sending error messages to the terminal, the virtual operator informs a designated person of the errors through electronic mail. See "Using Unattended Operations" on page 4-50 and "The onautovop Utility" on page 9-9 for more information.

- Emergency situations

  An emergency situation exists when Universal Server cannot access the ON-Archive catalog, thus prohibiting ON-Archive from functioning. In these situations, an operator uses a special-purpose program, **ondatartr**, to retrieve data from tapes, disk, or backed-up logical-log files. See "Emergency Situations" on page 4-48 and "The ondatartr Utility" on page 9-12 for more information on emergency situations.

Universal Data for Archives and Backups

For a description of the Universal Server data that ON-Archive archives and backs up, see "What Is an Archive?" on page 1-4 and "What Is a Logical-Log Backup?" on page 1-4.

ON-Archive Commands and Requests

You perform an ON-Archive operation when you issue a command. A command consists of the command name and qualifiers, separated with the slash character (/). For example, the following ARCHIVE command requests to archive all Universal Server dbspaces:

```
ARCHIVE/DBSPACESETS=*
```

ON-Archive provides both a command-line interface and a menu interface for submitting commands. See "Using ON-Archive Commands" on page 4-7 for a complete list of ON-Archive commands. See "Using the ON-Archive Command-Line Interface" on page 4-16 and "Using the ON-Archive Menu Interface" on page 4-17 for information on how to submit an ON-Archive command.

ON-Archive Requests

The ON-Archive commands that perform archive, backup, and restore operations are distinguished from other commands because they do not execute immediately. These special commands, called *requests*, are stored in the ON-Archive catalog to be executed at a later time. Requests are executed either using an operator instruction (the EXECUTE command) or using the virtual-operator program, **onautovop**, that manages unattended operations. See "The EXECUTE Command" on page 4-27 for information on how to explicitly execute a request. See "Using Unattended Operations" on page 4-50 for information on how to use unattended operations.

Figure 2-2 illustrates the two-step operation of creating and explicitly executing a request to create an archive. In the first step, you enter an ARCHIVE *command*, that ON-Archive processes and stores in the catalog. Once ON-Archive stores that command in the catalog, the ARCHIVE command becomes a *request*. In the second step, you issue the EXECUTE command to explicitly execute the request and start the archive.

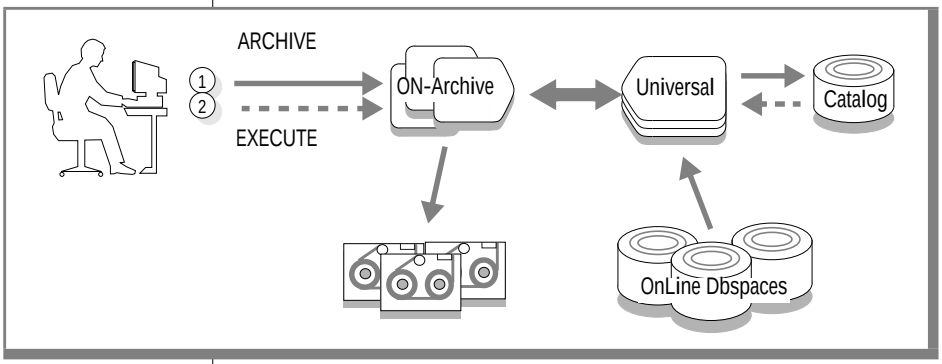


Figure 2-2
The Two-Step Operation of Creating and Executing a Request

See "Creating a Request" on page 4-26 for a list of the ON-Archive commands that become requests.

The ON-Archive Programs

This section describes the programs that make up ON-Archive.

onarchive

The **onarchive** program enables you to enter ON-Archive commands. It provides both a command-line interface and a menu interface. Both interfaces provide the same ON-Archive functionality.

See Chapter 4, "Using ON-Archive," for information on how to start **onarchive** and how to invoke the command-line interface or the menu interface.

oncatalgr

The **oncatalgr** program is a daemon-like process that ON-Archive uses to access the catalog. The **oncatalgr** process maintains an open connection to the **sysmaster** database, where the ON-Archive catalog tables reside. See Chapter 10, "The ON-Archive Catalog," for a complete description of the ON-Archive catalog.

The **oncatalgr** process must be running to use ON-Archive for attended or unattended operations. Thus, choose wisely and start **oncatalgr** when you bring Universal Server on-line and stop it when you take Universal Server off-line. See "Starting and Stopping the Cataloger" on page 4-5 for instructions on how to start and stop **oncatalgr**.

onautovop

The **onautovop** program acts like a virtual operator. It executes scheduled ON-Archive requests without requiring human intervention. When **onautovop** executes requests, they are called *unattended operations*. See "Using Unattended Operations" on page 4-50 and "The onautovop Utility" on page 9-9 for more information about **onautovop**.

ondatartr

The **ondatartr** utility, described in "The ondatartr Utility" on page 9-12, allows **root** or **informix** to perform ON-Archive operations in emergency situations when you cannot use ON-Archive because the catalog is not available. The **ondatartr** utility uses a command language similar to that of **onarchive** but has no menu interface. See "The ondatartr Utility" on page 9-12 for more information about **ondatartr** and its command language. Also see "Emergency Situations" on page 4-48.

The ON-Archive Catalog

The ON-Archive *catalog* is central to ON-Archive operations. The ON-Archive catalog is a set of Universal Server database tables that are part of the **sysmaster** database. Unlike the system-monitoring-interface tables in the **sysmaster** database, the ON-Archive catalog tables are stored on disk, like regular Universal Server tables.

The ON-Archive catalog maintains information on ON-Archive data and operations. For example, the ON-Archive catalog tracks the following items:

- Definitions of volume sets and volumes
- Definitions of dbspace sets
- A record of archive and backup requests
- Locations of tape volumes

See "What Are Volume Sets and Volumes?" on page 2-9 for a description of volume sets and volumes. See "What Is a Dbspace Set?" on page 2-19 for a description of dbspaces. ON-Archive automatically updates the catalog when you execute commands and requests.

See Chapter 10, "The ON-Archive Catalog," for a complete description of the ON-Archive catalog tables.

How ON-Archive Organizes Data

ON-Archive uses the following categories to organize Universal Server archive and backup data:

- *Volume sets* and *volumes*, that refer to the physical media that ON-Archive uses for archives and logical-log backups
- *Save sets*, that refer to the logical grouping of data on volumes
- *Dbspace sets*, that identify groups of Universal Server dbspaces that you can archive as a set

This section describes each of these data sets. You use ON-Archive commands to create volume sets, volumes, and dbspace sets. ON-Archive creates save sets automatically when it performs archives and logical-log backups.

What Are Volume Sets and Volumes?

ON-Archive organizes the physical media that it uses to store back-up data (archived data and backed up logical-log files) into *volumes sets* and *volumes*.

Volumes must always belong to a volume set; therefore, you must define a volume set before you define any volumes. Volumes in a volume set share the same characteristics, such as type (disk or tape), density, and so on. In a sense, volumes in a vset make up a logical volume. ON-Archive manages the individual volumes within a volume set, so back-up data can span multiple volumes without causing any special problems.

The following sections describe volume sets and volumes in more detail.

What Is a Volume Set?

A *volume set* (or *vset*) consists of a group of tapes or disk directories that you define as a repository for your archive and logical-log backup data (see Figure 2-3). Your ON-Archive system must contain at least one vset. "Using Volume Sets and Volumes" on page 4-31 explains how to create a vset.

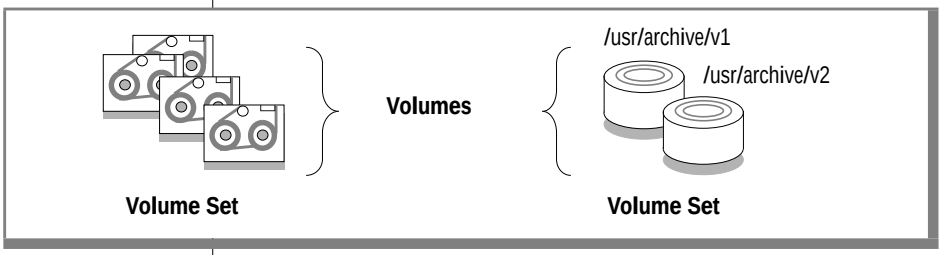


Figure 2-3
Volume Sets are Made Up of Volumes

You can create an unlimited number of vsets. When you define a vset, however, you might want to consider the following purposes:

- Grouping data that serves a common purpose—the databases for a particular department, for example
- Grouping data from a certain point in time—archive and logical logs for a specific day in the backup schedule, for example

When you group data having a common purpose in the same vset, you can easily restore only that subset of Universal Server data when the need arises. Grouping data that serves a common purpose enables you to manage separately the archive and backup requirements for different groups, departments, applications, or projects. As an alternative, you could group your databases according to the value your organization places upon them, then define a vset for each group and archive it separately.

You must back up all logical-log files on the same device type or you cannot perform a restore. For this reason, you might want to put logical-log backup data on a different vset than archive data. For that same reason, you might also consider storing logical-log backups on a single vset.

When you choose to keep data from a certain point in time together, you might define vsets based on your archive schedule. For example, you could define three vsets called **Rotation1**, **Rotation2**, and **Rotation3**. When you put Monday's archive on **Rotation1**, Wednesday's archive on **Rotation2**, and Friday's archive on **Rotation3**, you could start over with **Rotation1** on Monday again. This archive schedule ensures that you always hold three rotations worth of archive data.

You could also define vsets based on both content and time distinctions. For example, you might define a vset for each day in the archive schedule and define another vset for logical-log backups.

What Is an Imported Volume Set?

Sometimes you might need to read data that was created with a different ON-Archive system. You might want to protect that vset from being written to or copied, ensuring that it can only be read or restored. In cases like this, you can define the vset as *imported.* Once you properly catalog the vset, you can retrieve the content of an imported vset. However, you cannot archive or copy new data to it, and you cannot copy the contents of an imported vset to another vset.

Tip: *When you prepare to use Universal Server data replication and you create an archive to replicate the data for the first time, you must define the volumes in the vset as imported to protect them.*

What Is a Transit Volume Set?

A *transit vset* consists of a special vset that ON-Archive can use to store data temporarily when an archive or logical-log backup overflows a tape vset during an unattended operation. After the operation completes, you must move the data to a permanent volume set. Although the data stored in a transit vset resides there temporarily, the transit vset is a *permanently mounted* disk vset that ON-Archive uses when it cannot find an available tape volume. When you specify the TRANSIT qualifier for an archive or backup to a disk vset, ON-Archive displays an error message and terminates the request with a status of FAILED.

You cannot archive or copy data directly to a transit vset. ON-Archive uses a transit vset in the following instances:

- When data overflows a volume during unattended operations

  When you execute an archive or logical-log backup to tape, a transit vset eliminates the need for an operator to be physically present in case the data overflows a volume. Use of a transit vset permits the overflow data to be stored on disk and migrated to tape the following morning. (See "Using Unattended Operations" on page 4-50.)

- When a storage device is unavailable during unattended operations

  When a storage device is off-line or specifically allocated to another task, use of a transit vset permits **onautovop** to continue uninterrupted. When the storage device becomes available, you can transfer the data from the transit vset.

The transit vset is used only under the following conditions:

- A request specifies the TRANSIT qualifier.
- The **onautovop** utility executes the request.
- No volume is available in the vset that the request specified.

When **onautovop** uses a transit vset, it generates a COPY request to copy the data it stores in the transit vset to the vset specified in the request. You cannot use data in a transit vset for a restore until you copy it to a regular vset. You must not consider the data truly safe until then.

When ON-Archive uses a transit vset, the status of the initial request changes to PARENT. The copy request is referred to as a CHILD request and it has a status of NEW. You can execute it using **onautovop**, or manually using **onarchive**. When the child-copy request is complete, the status of the original request changes to COMPLETED. To execute the COPY request, you must mount a tape volume on the designated tape device.

When the status of a child-copy request is CANCELLED or FAILED, the status of its parent changes to UNCOMPLETED.

What Is a Volume?

An ON-Archive *volume* is either a single tape or a single disk directory that you use to store Universal Server archive or logical-log backup data. It is one unit of a volume set, as shown in Figure 2-4.

Pay no concern to the characteristics of the storage device when you define a volume. You define the characteristics of particular storage devices in the ON-Archive configuration file. Then, when you define a *volume set*, you assign it to a device type—either disk or tape.

Figure 2-4
Tape and Disk Volumes

Warning: *When you use disk volumes to store your archive and backup data and you experience a disk failure that requires you to restore your Universal Server data, your archive data could also be inaccessible due to the failure. When you use disk volumes for archives or backups, as a minimal precaution you must define the volumes on disks that Universal Server does not use.*

Tip: *Use disk volumes to speed up archives and backups. When the operation completes, use the COPY/VSET command to copy the volumes to tape for safety. After you copy the data to tape, use the REMOVE/REQUEST command to free up the disk space.*

A volume is not expandable in the sense that the capacity of the physical medium limits the storage space that the volume provides. When an archive or logical-log backup is too big to fit onto one disk or tape volume, ON-Archive automatically extends the archive or backup onto another volume. See "The DEFINE/VOLUME Command" on page 4-38 for information on how to create a volume.

What Is a Volume Label?

When you define a volume, ON-Archive issues it an electronic label. The label identifies the volume and its contents, and prevents ON-Archive from accidentally overwriting the contents of a volume that might be needed for a restore.

For tape volumes, ON-Archive writes the volume label to the tape. For disk volumes, the volume header is a file called **VOL.HDR** in the volume directory.

Can Data Be Placed on a Specific Volume?

When you create a request to archive data or back up your logical-log files, you associate that request with a *volume set*, not a particular volume. For example, the following request backs up logical-logs to a volume set called **log_vset**:

```
BACKUP/LOGFILE/AUTOMATIC/VSET=log_vset
```

You cannot specify that the log backup be made to a specific tape within the volume set; ON-Archive automatically selects a volume from the volume-set when an archive or backup command is executed. Which volume ON-Archive uses is explained in "How ON-Archive Selects Volume Sets, Devices, and Volumes" on page 4-46.

Although you cannot place data on a specific volume, you can specify that a request be created on a blank volume, and that no other requests use that volume, as explained in "Keeping Archives on Separate Volumes" on page 5-17.

Volumes in a Disk-Volume Set

For tape volumes, many volumes often reside in a vset; but for disk volumes, no reason exists for more than one volume to reside in a vset. When you define your disk volume as large enough, no advantage exists to having multiple disk volumes in a vset.

What Is a Save Set?

A *save set* is a logical grouping of archive or logical-log backup data within a volume set. ON-Archive automatically creates a save set for each of the following sets of data:

- All the data archived using a single archive request
- The logical-log files backed up using an automatic logical-log-backup request
- Each logical-log file backed up as part of a continuous logical-log backup request
- The logical-log files that **ondatartr** backs up for an emergency backup or to salvage logical-log files

In addition to archive and logical-log backup data, a save set also contains general information such as the ON-Archive or **ondatartr** request for an archive or logical-log backup, the name of the person who created the request, the time the request was executed, the name of the operating system, whether the data is compressed or encrypted, and so on.

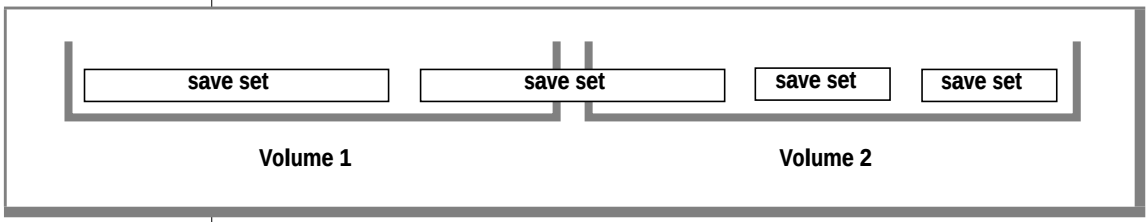
Each item within the save set (a logical-log file or dbspace, for example) is identified using a file header enabling ON-Archive to selectively retrieve individual items even though they are stored in one save set.

Each save set is identified using a save-set id, that represents the request id of the archive or logical-log backup request. For save sets created by **ondatartr**, the save-set id is the operating-system process id of the **ondatartr** process.

Save Sets and Tape Volumes

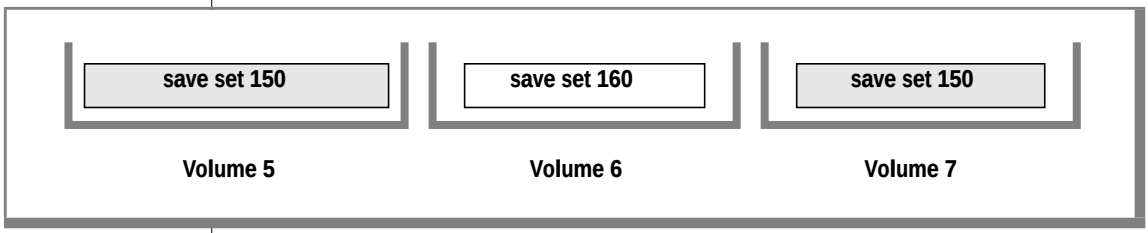
As illustrated in Figure 2-5, save sets can span volumes. Also, a single volume can contain more than one save set. However, you can force save sets to reside on separate volumes, when that is easier for you administratively, as described in "Keeping Archives on Separate Volumes" on page 5-17.

Figure 2-5
Save Sets and Volumes



When several concurrent requests execute using the same vset, it is possible for a a save set to reside on nonsequential volumes. For example, consider the following scenario. Request 150 uses a vset that is also used by request 160. Request 150 executes and uses volume 5 in the vset. Request 160 executes shortly after that. Because volume 5 is being used by request 150, request 160 uses volume 6. When request 150 runs out of room on volume 5 and request 160 is still using volume 6, request 150 finishes its save set on volume 7. As shown in Figure 2-6, save set 150 is on volumes 5 and 7.

Figure 2-6
Save Sets Can Span Nonsequential Volumes



Save Sets and Disk Volumes

Although save sets on disk can also span volumes, as Figure 2-5 on page 2-16 illustrates, the previous scenario is more representative of tape volumes than disk volumes because multiple requests can write to the same disk directory without interfering with each other. For disk volumes, a save set is an individual file within the volume directory. The name of the file contains the save-set identification number. For example, when you use your operating system to list the contents of a disk volume called **/archive/arc_vset/vol** that contains four save sets, you see something like this:

```
% ls /archive/arc_vset/vol
VOL.HDR
00000001.SAV
00000002.SAV
00000003.SAV
```

What Devices Does ON-Archive Use?

An ON-Archive *device* is a tape device or disk directory that is defined to the operating system and that ON-Archive can use to read from and write to a volume. ON-Archive defines a device as both a particular *physical device*, such as `/dev/rst0` (tape) or `/usr/archive/monday` (disk), and a *logical device*, or device type—that is, tape or disk. The following sections explain how ON-Archive defines and uses physical and logical devices.

The Physical Device

For ON-Archive, you can specify a physical device as a tape drive or a disk. For more information on the specific physical devices that ON-Archive can use, see “DEVICE” on page 3-8.

The Logical Device

A *logical device* identifies a group of one or more physical devices of the same type. You define a logical device using the ON-Archive DEVICE configuration parameter. Using the DEVICE parameter, you specify a logical device name, followed by an equal (=) sign and a list of the physical devices you want to assign to that logical device. For example, the following DEVICE parameter defines a logical device called **tape** and associates the physical devices **/dev/tapedev1** and **/dev/tapedev2** with it:

```
DEVICE tape = /dev/tapedev1, /dev/tapedev2
```

When you request an archive or logical-log backup, you specify the name of the volume set to which you write the data. Furthermore, you specify the logical device that ON-Archive uses because you assigned the volume set to a logical device when you defined it. When ON-Archive executes the archive or backup, it prompts you to select one of the physical devices defined for that logical device.

See Chapter 3, "Configuring ON-Archive," for a complete description of the DEVICE parameter and more information on defining logical and physical devices. See "The DEFINE/VSET Command" on page 4-31 for information on assigning a logical device to a volume set.

What Is a Dbspace Set?

A *dbspace set* is one or more Universal Server dbspaces that you designate as a unit. Defining a dbspace set enables you to archive multiple dbspaces using a single name for the set. You use the DEFINE / DBSPACESET command, described in "The DEFINE / DBSPACESET Command" on page 5-5, to define a dbspace set. Figure 2-7 illustrates two dbspace sets.

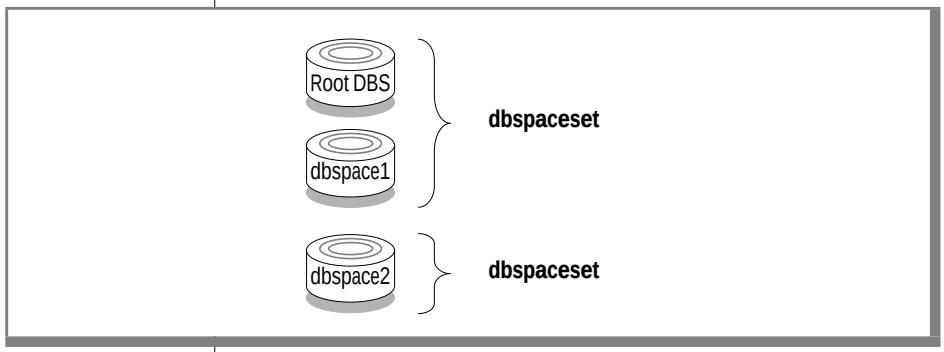


Figure 2-7
Dbspace Sets

You might want to archive your data in dbspace sets for either or both of the following reasons:

- Perform operations in parallel to decrease the time it takes to archive and restore
- To recover efficiently from a single disk failure

The following two sections explain these reasons.

Dbspace Sets Can Enhance Parallelism

You can divide your dbspaces into dbspace sets to increase the degree of parallelism that you can use to archive or restore. You might want to divide your data based on the size of the dbspaces, the frequency that the data in the dbspaces is updated, or both.

For example, when frequently updated tables reside in some dbspaces, and some infrequently updated tables reside in other dbspaces, you might want to define a dbspace set that contains the frequently updated dbspaces, and one that contains the relatively static dbspaces. Then, you might only need to archive the static dbspace set once a week, while the frequently updated dbspace set might need archiving every other day.

When one large dbspace exists with several small ones, you could place the large dbspace in one set and all the small ones in another. That way you could archive the large dbspace as one set on one device and simultaneously archive all the other dbspaces as another set on a different device.

Dbspace Sets Can Speed Recovery After a Disk Failure

When a disk with Universal Server data fails, you only need to restore the dbspaces that use chunks on that disk. When the disk that failed does not contain chunks for the root dbspace, or the dbspaces containing the logical log or the physical log, Universal Server can remain on-line—keeping other dbspaces available—while you restore the dbspaces that were lost. When the disk does contain chunks for the root dbspace, or the dbspaces that contain the logical log or the physical log, Universal Server comes off-line when the disk fails and you must perform the restore while Universal Server is in off-line mode. See "A Full-System Restore" on page 7-5 for more information on restoring Universal Server data while Universal Server is in off-line mode.

When you group dbspaces, that use chunks on the same disk into a single dbspace set, you can restore the disk in a much easier fashion; you can restore all of the dbspaces in a single physical restore. When two dbspaces in different dbspace sets use chunks on the same disk, you must perform two physical restore operations to restore the disk, one for each dbspace or dbspace set. Furthermore, you can always restore an individual dbspace from a dbspace set, or even from a full-system archive.

Rules for Dbspace Sets

Keep in mind the following rules when you create dbspace sets:

- A single dbspace can reside in more than one dbspace set.
- No dbspace must belong to any dbspace sets; however, when a dbspace is not archived, you cannot restore it.

You do not have to use dbspace sets; you can create a full-system archive (except for temporary dbspaces) without defining a dbspace set. See "Making a Full-System Archive" on page 5-16.

Concurrent Archives and Restores

ON-Archive enables you to perform multiple archives, backups, or physical restores simultaneously. You can execute several requests at the same time to decrease the time required for archives or restores for this feature. You can perform the following types of parallel activities:

- Execute several requests to create archives of different dbspace sets at one time
- Restore from multiple save sets simultaneously
- Create archives while restoring data from one or more previous archives.

ON-Archive has some limitations on the amount of concurrency you can achieve, based on the available devices and the requests you want to execute.

The **oncatalgr** process can support 99 instances of **onarchive** and **onautovop**; in other words, each instance of Universal Server can support 99 concurrent archives.

Device Configurations That Increase Concurrency

The following three device configurations allow you to execute multiple requests simultaneously:

- Multiple tape drives
- Permanently mounted disks
- Removable disks

ON-Archive considers that removable disks and multiple tape drives are the same with respect to concurrent request execution.

Disk-Space Allocation During Concurrent Operations

When multiple archive, backup, or copy requests use the same disk volume, each operation must not exceed the limit on the size of the disk volume imposed by the MAX_SPACE qualifier. Because more than one set of data is being written to the volume, the amount of space remaining for any one operation depends on the amount of space other operations are taking. To keep the operations from continually checking how much space is remaining in the volume, space is allocated to each operation in the amount of BLOCKSIZE (specified in the request or operator default file) multiplied by the NB_DISK_SPACE_EXTENT parameter defined in the ON-Archive configuration file, **config.arc**. When a save set uses all the space in one (BLOCKSIZE * NB_DISK_SPACE_EXTENT) allocation, it is given another allocation of the same size.

Tape Reliability and Storage Options

ON-Archive offers the following features that increase tape reliability:

- ON-Archive can create copies of save sets to prevent against media failure.
- ON-Archive can add a cyclic redundancy check to archives and backups.
- ON-Archive prevents overwriting tapes.
- ON-Archive can compress data as it is written to a save set.

Each of these features is explained in the following sections.

Copies of Save Sets

ON-Archive can make copies of a particular archive save set. Then, in the event of a media failure, you can retrieve the archived data from one of the copies. You also might want to create copies for off-site storage at a different facility. See "Creating Multiple Copies of an Archive" on page 5-17.

As explained in "Using Unattended Operations" on page 4-50, Informix recommends that you create copies of logical-log files when you back them up.

You cannot copy to or from an imported vset.

Cyclic-Redundancy Check

You can use the CRC qualifier when you create archives and backups to instruct ON-Archive to add a cyclic-redundancy-check (CRC) field at the end of each save-set block. The CRC field stores the result of an algorithm that ON-Archive performs on the data in the block at the time it writes it. When ON-Archive later reads the block, it performs the same algorithm on the data again and compares the result to the value in the CRC field. When they do not match, ON-Archive knows that the data in the save set is corrupt.

Tape-Overwrite Protection

The **onautovop** program always checks volumes before it writes to them and does not use volumes that are not correct for the request. While **onarchive** interacts with the operator when a volume is not the one it expects, **onautovop** simply does not use the volume.

You must predefine volumes so that ON-Archive can use tape volumes without operator intervention. (See "The DEFINE / VOLUME Command" on page 4-38.) To define volumes, you must mount the tape on the correct device so that ON-Archive can write the vset name, volume header, and other identifying information to the media. This procedure might seem like an inconvenience, but it keeps ON-Archive from overwriting potentially valuable information.

Data Compression

Universal Server can compress data as part of an archive or backup operation. Compression is a transformation applied to the input data to reduce the size of the archive before ON-Archive sends the data to the archive device. Compression is reversible; you can reverse the compression process to restore the original data from the archived data.

Data compression reduces storage requirements but imposes processing overhead. (You must decompress data during retrieval.) Whether data compression is worthwhile depends on how important the extra time spent creating the archive is and the quality of the compression ratio. The compression ratio depends on the method used, and the nature of the data designated for archiving.

The compression method provided with ON-Archive is called REP. It is a very quick method that requires minimal overhead. REP eliminates streams of identical bytes to achieve compression. This method works best with text data and allows a compression ratio of 30 to 40 percent.

Contact Informix when you wish to use a compression method other than REP.

ON-Archive Security

ON-Archive provides the following security features:

- Controlling user access to ON-Archive using privilege modes

  You can restrict the use of archive-and-restore facilities to operators (**root** and **informix** accounts) or to a group of users you specify, or you can designate all users to use archive-and-restore facilities.

- Controlling access to storage

  ON-Archive allows you to define volumes and vsets so that specified users can only perform certain actions on them.

- Data encryption

  ON-Archive provides data encryption to encrypt data before it is written to a volume.

Each of these features is explained in the following sections.

Privilege Modes

Privilege modes affect how ON-Archive commands work, depending on the user that executes the command. ON-Archive runs in at least one of two *privilege modes* at all times. It also can run in both privilege modes.

The following list shows the ON-Archive privilege modes:

- OPERATOR
- GROUP

The privilege mode for ON-Archive is set in the ON-Archive configuration file, **config.arc**, as described in "PRIVILEGE" on page 3-17. The following sections describe each of the three privilege modes.

OPERATOR Privilege Mode

OPERATOR privilege mode allows only users **informix** and **root** to perform activities that impact the ON-Archive system (creating archives, doing restores, defining volumes, and so on). This mode is the default. It is useful because it protects the nonadministrative users of your ON-Archive system from making potentially costly mistakes.

GROUP Privilege Mode

GROUP privilege mode confers on other users the privileges that **informix** and **root** experience in OPERATOR privilege mode. It allows you to specify an operating-system group of users you trust to perform activities that impact the ON-Archive system (creating archives, doing restores, defining volumes, and so on) so that ON-Archive recognizes that group.

GROUP privilege mode has the advantage of OPERATOR mode in that it protects nonadministrative users from making uninformed errors. It also offers flexibility because it lets you designate the accounts that are allowed to perform administrative tasks. (You might want to include **informix** and **root** in the group, or run in both GROUP and OPERATOR privilege mode.)

Creating the operating-system group for GROUP privilege mode is explained in "Creating a super_archive Group" on page 3-19.

Controlled Access to Storage

To control access to storage (volumes and vsets), ON-Archive uses a combination of access permission and access-control lists.

Access-Control Lists

When you define vsets, you use the CLASS qualifier to assign either the SYSTEM or USER class to them. Vsets that you define with the SYSTEM class are available to any user on the system, according to the access permissions defined for that vset.

Vsets that you define with the USER class are available only to users on the access list for that vset. The access-control list is defined when you create a vset with the USER qualifier and can be modified at any time. All users on the access list possess the same access permissions defined for that vset.

When ON-Archive is in OPERATOR mode, **root** and **informix** can always access all volumes and vsets, subject to the PROTECTION qualifier, regardless of whether they are of the SYSTEM or USER class. In GROUP mode, users in the **super_archive** group must be listed in the list of USERS for the vset to gain access to it.

Access Permission

Access permissions determine what types of accesses are allowed on a volume set or volume. The following list shows the types of access permissions available:

- R (read)
- W (write)
- D (delete)

The same type of access permissions (R, W, or D) can be granted to both volumes and vsets. Access for both vsets and volumes is specified at the time they are defined using the PROTECTION qualifier. They can be modified at a later time using the MODIFY/VSET and MODIFY/VOLUME commands.

Read-Access Permission

Read access allows any read operation from the physical media (volume). When read access is not granted on a vset or volume, no read operations are allowed; you cannot retrieve data from it, nor can you copy data to another vset. You can list vset and volume definitions without read access, but you cannot access the stored data.

Write-Access Permission

Write access allows all write operations to the physical media (volume). When write access is not granted on a vset or volume, you cannot archive or copy data onto it. In addition, you cannot change the vset or volume definitions without write access.

Delete-Access Permission

Delete access allows you to delete a vset or volume definition. When delete access is not granted, you cannot delete a vset. Also, you cannot delete a vset when one of its volumes lacks delete access.

You can only delete volumes when no requests exist in the ON-Archive catalog that writes to the volume. When you cannot delete one of the volumes in a vset because a request could write to it, you cannot delete the vset containing that volume.

When a volume definition is deleted from the catalog, the information describing the contents of the volume is also deleted from the catalog. The deletion of a vset automatically causes the deletion of all its member volumes.

Volume Set and Volume Protection

Because volumes are always a part of a vset, certain rules describe how access permissions on vsets and volumes interact.

Figure 2-8 on page 2-29 shows that when a vset is restricted from certain types of access, all its volumes are similarly restricted.

Figure 2-9 on page 2-29 shows that a volume might have specific access restrictions beyond those associated with the vset to which it belongs.

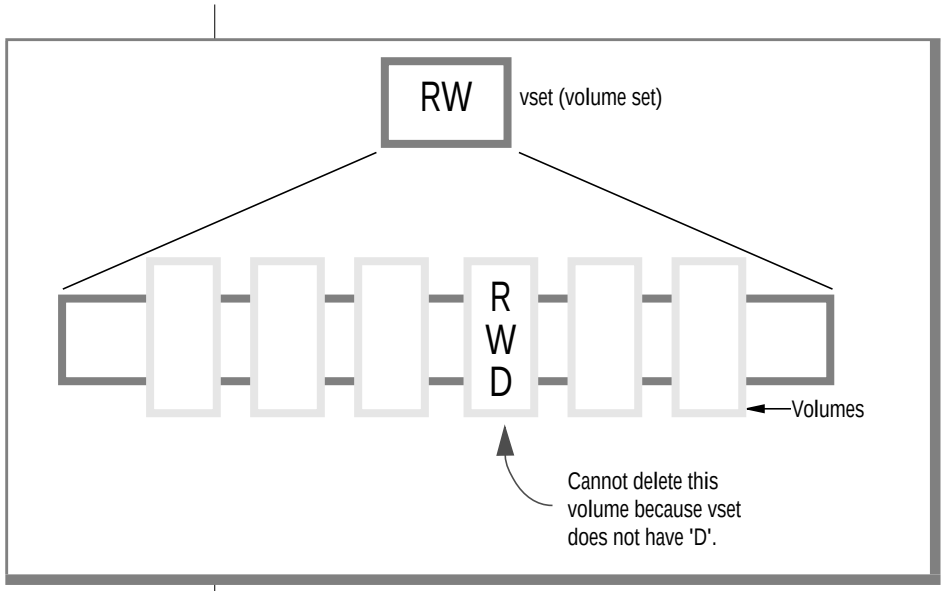


Figure 2-8
Vset Protection Can Override Volume Protection

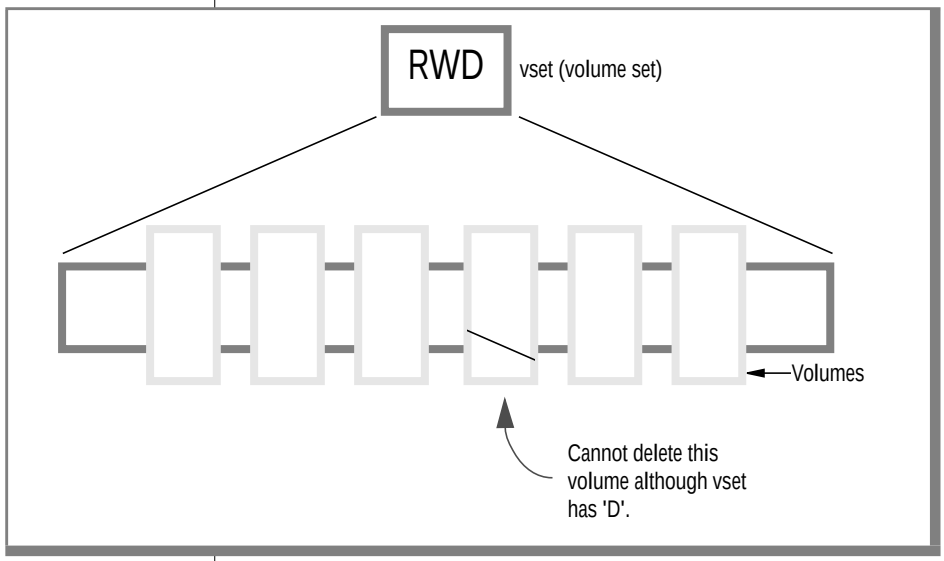


Figure 2-9
Volume Protection Can be More Restrictive Than vset Protection

When no protection is specified when a vset or volume is defined, the vset or volume is assigned the default access types as defined in the command qualifier default file. (See "Which Defaults Are Used?" on page 4-14.)

Data Encryption

ON-Archive can encrypt data as part of an archive or backup operation. Encryption is a transformation applied to the input data to render it incomprehensible before sending it to the archive or backup device. Encryption is reversible. In other words, you can reverse the encryption process to restore the original data from the archived or backed-up data.

ON-Archive can encrypt the data as it is archived, but it does not do so automatically; you must specifically request encryption for an archive. ON-Archive uses the XOR (exclusive OR) encryption method. It incurs relatively little overhead and is sufficient to prevent the curious from reading the data. You must contact Informix when you wish to use an encryption method other than XOR.

To encrypt data in a secure manner, ON-Archive uses the following three types of passwords, also called *keys*:

- User-encryption method and key (UEMK)
- System-encryption key (SEK)
- Internal key

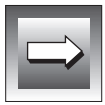
Each key type is explained in the following sections.

What Are User-Encryption Keys?

Users can use the ENCRYPT qualifier, when they create a request, to specify an encryption method for an archive or backup. They supply an *encryption key* and *encryption method* with the ENCRYPT qualifier. The encryption key and method— called the *user-encryption method and key* (UEMK)— is similar to a password; the user must also specify the same key and method with the DECRYPT qualifier to decrypt the data when it is retrieved.

Users can specify a different UEMK for each archive operation. Once the request has been executed, you cannot change the UEMK. To modify the UEMK for a pending request, you can use the MODIFY command. See "The MODIFY / COMMAND Command" on page 8-11.

When you need to find out a forgotten UEMK, the ON-Archive administrator can use the **onkeymgr** utility. See "The onkeymgr Utility" on page 9-29 for information on how to use the **onkeymgr** utility.



Important: *Do not include default encryption parameters in the default qualifier file,* ***oper_deflt.arc****. This creates a security risk. See "Where Qualifier Default Values Are Specified" on page 4-13 for information on* ***oper_deflt.arc*** *file*

What Is the System-Encryption Key?

When the request that uses encryption is stored in the catalog, the UEMK is also stored in the catalog. When it is stored, the UEMK is encrypted using the *system-encryption key* (SEK) to prevent other users from learning the key. There is only one SEK for a database server. Before using data encryption, the ON-Archive administrator must change the supplied SEK using the keymanager utility.

To change the SEK, the ON-Archive administrator runs the **onkeymgr** utility. See "The onkeymgr Utility" on page 9-29 for more information on the **onkeymgr** utility. Previously encrypted UEMKs in the catalog are automatically reencrypted with the new SEK.

Only the ON-Archive administrator must know about the SEK. And you must document the SEK in a secure fashion.

The default SEK delivered with ON-Archive is KARCHIVE.

Important: *Before using the encryption features, change the default SEK. Using the default SEK creates a security risk.*

What Is an Internal Key?

In addition to being stored in the catalog, the UEMK (not the SEK) is stored together with the data it encrypted as part of the save sets. When you must rebuild the catalog in the case of disaster recovery, the CATALOG command can retrieve the UEMK in each save set.

Just as the SEK encrypts the UEMK when stored with a request in the catalog, the UEMK stored in the save set is also encrypted. Instead of the SEK encrypting UEMK, a special *internal* ON-Archive key encrypts the UEMK. This internal key is common for all ON-Archive installations.

The fact that the keys are common enables ON-Archive to fully catalog save sets (including the UEMK) in imported vsets that contain encrypted data.

Chapter

3

Configuring ON-Archive

This chapter describes the ON-Archive configuration parameters. It also describes how to accomplish the following tasks:

- Change ON-Archive configuration parameters
- Create a **super_archive** group to manage security

ON-Archive Configuration Parameters

The default configuration file for ON-Archive is the **config.arc** file, located in the **$INFORMIXDIR/etc** directory.

When you wish to create and use a different configuration file for ON-Archive, you can set the **ARC_CONFIG** environment variable to the name of the file. and Universal Server looks for the file specified by **ARC_CONFIG** in the **$INFORMIXDIR/etc** directory. The **ARC_CONFIG** environment variable enables you to change configuration parameters while preserving the default **config.arc** file. It also enables you to create multiple configuration files and select one by setting **ARC_CONFIG**.

ON-Archive also provides a file named **oper_deflt.arc** that defines default values for command qualifiers. See "DEFAULT" on page 3-7 and "Qualifier Default Values" on page 4-13 for more information on this file.

Warning: *When you install ON-Archive, the install script checks the* ***$INFORMIXDIR/etc*** *directory for the existence of the* ***config.arc*** *and* ***oper_deflt.arc*** *files. When they exist, the script does not write over them; it installs the new versions of these files as* ***Config.arc*** *and* ***Oper_deflt.arc****. After the installation is complete, and your* ***config.arc*** *and* ***oper_deflt.arc*** *files already exist, you must compare them to the* ***Config.arc*** *and* ***Oper_deflt.arc*** *files to determine whether new configuration or qualifier parameters exist. When new parameters exist, you must add them to the* ***config.arc*** *and* ***oper_deflt.arc*** *files because ON-Archive uses these files. As an alternative, when you want to accept the ON-Archive default values in these files, you can simply copy the* ***Config.arc*** *and* ***Oper_deflt.arc*** *files to* ***config.arc*** *and* ***oper_deflt.arc****, respectively.*

The following table defines the parameters in the ON-Archive configuration file:

CATALOG MESSAGE	specifies the **oncatlgr** message file.
DEVICE	specifies the devices that ON-Archive can use to read and write archives and logical-log backups.
DEFAULT	specifies the name of the file that contains defaults for command qualifiers.
ENGLISH	indicates that the default language is English.
ERROR	specifies the names of the error-message files that ON-Archive uses.
HELP	specifies the name of the **onarchive** Help file.
KEYM_HELP	specifies the **onkeymgr** Help files.
MESSAGE	specifies the names of the format files.
NB_DISK_SPACE_ EXTENT	defines the unit of space allocation on disk volumes.
PRIVILEGE	indicates the restrictions on ON-Archive use.
TIME_OUT	specifies a time-out value.

Default Values

Figure 3-1 shows the contents of **config.arc**, the default ON-Archive configuration file:

Figure 3-1
Contents of config.arc

```
ENGLISH

! Device available for onarchive.
DEVICE HO = /home
DEVICE ho = /home
DEVICE USR = /usr
DEVICE usr = /usr
DEVICE TAPE = /dev/rst0
DEVICE tape = /dev/rst0
DEVICE RTAPE = port_meirion:/dev/rst0
DEVICE rtape = port_meirion:/dev/rst0

! NOTE: all the following file names are relative to INFORMIXDIR
! Operator default file.
DEFAULT = /etc/oper_deflt.arc


! Error messages files.
ERROR ENGLISH = /msg/errmsg_E.dat

! Format files.
MESSAGE ENGLISH = /msg/fmt_E.dat

! Help files.
HELP ENGLISH = /msg/hlp_E.hpf

KEYM_HELP ENGLISH = /msg/hlp_km_E.hpf

! Catalog message file.
CATALOG MESSAGE = /msg/c_err_E.dat

! User privileges
PRIVILEGE = OPERATOR,
!PRIVILEGE = OPERATOR, GROUP

! Timeout value (in minutes)
TIME_OUT = 30

! Number of buffers allocated to write on DISK
! The size of a buffer is equal to /BLOCKSIZE used by Archive
and Copy commands.
NB_DISK_SPACE_EXTENT = 10
```

Syntax of Configuration Parameters

Parameter names in the ON-Archive configuration file appear in UPPERCASE letters. With one exception, the parameter name is followed by an equal sign (=) and a value. For example, the following line defines the location of the file, relative to **$INFORMIXDIR**, that specifies the default values for ON-Archive command qualifiers:

```
DEFAULT = /etc/oper_default.arc
```

The exception is the language parameter, because it lacks a value assigned to it.

You can specify only one parameter on each line of the file. The text for each parameter must not exceed 80 characters.

Lines that start with an exclamation point (!) or a hash mark (#) are comments.

Changing Configuration Parameters

To change the value of a configuration parameter, read the configuration file using a text editor, locate the parameter, change the value, and write the updated file to disk.

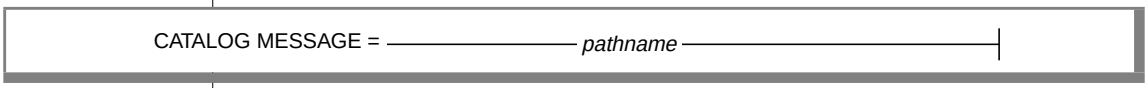
Important: *Anytime you modify the ON-Archive configuration file you must stop and restart the* ***oncatlgr*** *process to put the changes into effect. See* *"Starting and Stopping the Cataloger" on page 4-5* *for information on how to start and stop* ***oncatlgr****.*

The only ON-Archive configuration parameter that you *must* set is the DEVICE parameter. The DEVICE parameter identifies the physical devices that ON-Archive can write to when it creates archives and logical-log backups. You do not necessarily need to change the default values for the rest of the ON-Archive configuration parameters.

The following sections describe each of the ON-Archive configuration parameters.

CATALOG MESSAGE

The CATALOG MESSAGE parameter specifies the location of help information used by the **oncatlgr** utility. ON-Archive supplies the following help files for **oncatlgr**: **c_err_E.dat** (English version) and **c_err_F.dat** (French version):



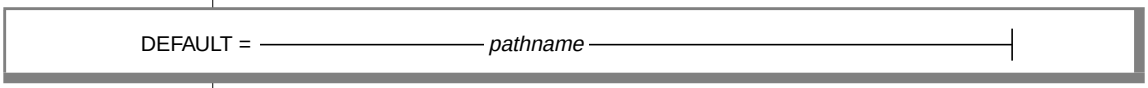
pathname is a filename relative to **$INFORMIXDIR**. The *pathname* must begin with a slash (/). The *pathname* specifies the location and name of the Help file for **oncatlgr**.

The following example illustrates the CATALOG MESSAGE parameter:

```
CATALOG MESSAGE = /msg/cm_english.dat
```

DEFAULT

The DEFAULT parameter specifies the operator command-qualifier default file. For more information on the operator default file, see "Qualifier Default Values" on page 4-13.



pathname is a filename relative to **$INFORMIXDIR**. The *pathname* must begin with a slash (/). The *pathname* specifies the location and name of the file that contains the qualifier default values.

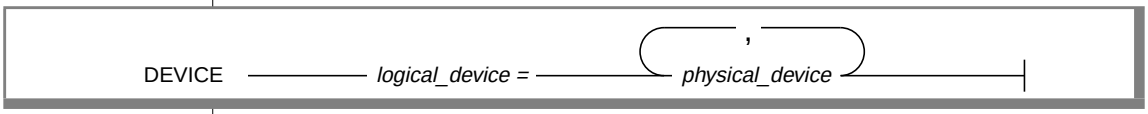
The following example illustrates the DEFAULT parameter:

```
DEFAULT = /etc/dflts.arc
```

DEVICE

A DEVICE parameter in the ON-Archive configuration file must define all physical devices that ON-Archive writes to or reads from.

The DEVICE parameter associates one or more physical devices with a logical device. (See "How ON-Archive Organizes Data" on page 2-9.) Assigning physical devices to a logical device enables you to group devices of the same type.



logical_device is a user-specified name that identifies a group of physical devices. Logical device names cannot exceed 10 characters.

physical_device is a physical-device specification associated with the logical-device name. The pathname to the physical device can be a link.

The DEVICE parameter supports the following tape devices. See the machine notes file, mentioned under "Documentation Notes, Release Notes, Machine Notes" on page -12 of the Introduction, to determine whether any restrictions apply to your platform:

- DAT (4-mm digital audio tape
- Exabyte (8-mm Helical scan)
- 9-track (half-inch reel to reel)
- QIC (quarter-inch cartridge)

When writing to tape devices, ON-Archive writes ANSI-standard tapes. Every volume and every save set on a volume contains standard header and trailer records limited to a length of 512 bytes. The block size that the user specifies with the BLOCKSIZE qualifier pertains to the size of the data blocks written after the headers are written. Thus, you can make the following assumptions:

- Because **onarchive** writes variable-length records, the tape device must be able to read and write, or be configured to read and write, variable-length records.
- The tape device must be able to handle records as small as 512 bytes.

The physical device might need to be rewound when it closes depending on whether it is a rewind or nonrewind device.

The physical device can be remote, as explained in "How ON-Archive Uses Physical and Logical Devices" on page 3-10.

The following example illustrates two DEVICE parameters, one that specifies a directory on disk and one that specifies two tape devices:

```
DEVICE USR      = /usr
DEVICE tape     = /dev/rst0, /dev/rst1
```

You can specify an unlimited number of DEVICE parameters in the configuration file.

Generally, when you experience a problem using **onarchive** with a tape device, the problem falls into one of the following categories:

- You are not using a driver for variable-length records where the size of the smallest record can be 512 bytes.
- The driver handles the end-of-data tape marks in an unexpected way.

You can fix the first problem by using a device driver for variable-length records (minimum size 512 bytes). The second problem requires a platform-specific fix to the **onarchive** I/O code.

Important: *The* ***onarchive*** *utility does not provide device drivers; either the platform manufacturer or the device manufacturer provides the device driver. ON-*Archive *does not control the behavior of the device driver. Any failures that result from incompatibility between* ***onarchive*** *and the device driver are not sporadic; they either work the first time or they do not. For a good test, create two or three small archives (like a level 0, 1, and 2 for a small dbspace) and put them on the same tape. Then, list the tape with* ***ondatartr*** *and do a restore from the tape. If this works, no incompatibility problem exists.*

How ON-Archive Uses Physical and Logical Devices

The DEVICE parameter groups physical devices of the same type under a logical device name. Grouping physical devices in this way enables you to direct the output of an operation to a *type* of device and select a specific device at the time of the operation. For example, the following DEVICE parameter groups two tape devices under the logical device name `tape`:

```
DEVICE tape = /dev/rst0, /dev/rst1
```

When you create a volume set, you assign it to a logical device through the DEVICE_TYPE entry on the DEFINE/VSET command. For example, the following DEFINE/VSET command defines a volume set named **arch** and assigns it to the logical device `tape`:

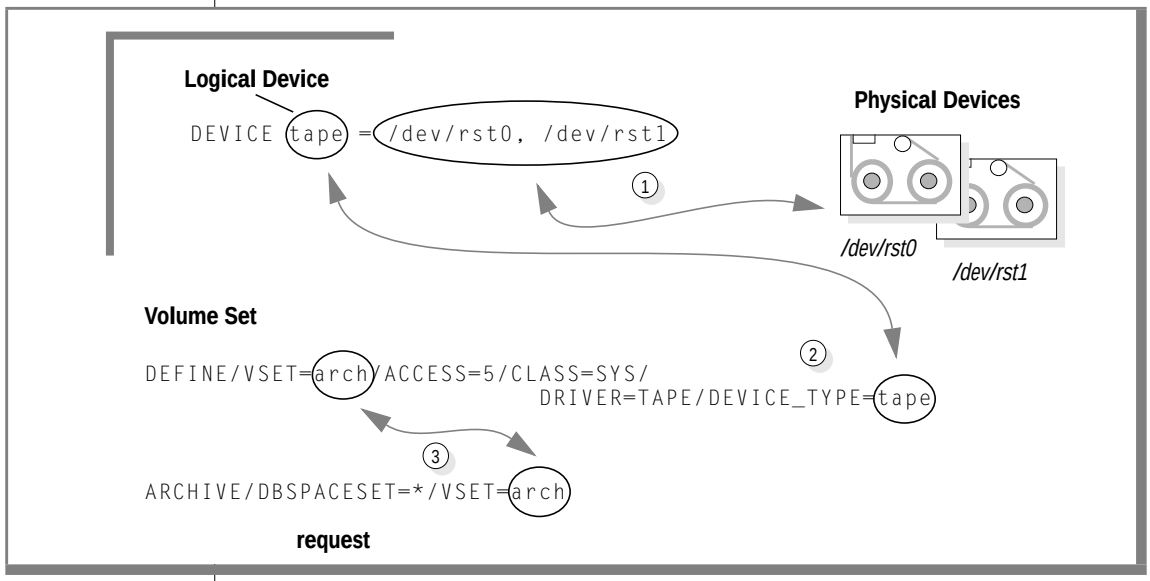
```
DEFINE/VSET=arch/ACCESS=5/CLASS=SYS/DRIVER=TAPE/DEVICE_TYPE=tape
```

When you enter a request to create an archive or a logical-log backup, you direct the output to a volume set and consequently to a logical device, or group of physical devices. For example, the following ARCHIVE command directs output to the volume set **arch**:

```
ARCHIVE/DBSPACESET=*/VSET=arch
```

Figure 3-2 illustrates how a request implicitly selects a group of physical devices by designating a vset.

Figure 3-2
How Requests are Associated with Devices



How ON-Archive selects a specific physical device depends on whether **onarchive** or **onautovop** executes the request.

When **onarchive** executes the request, it lists the physical devices assigned to the volume set by the device type (logical device) and allows you to choose one. Figure 3-3 illustrates how ON-Archive prompts you to select the physical device at the time it executes the request:

Figure 3-3
Devices Available for ON-Archive

```
Devices available for ONARCHIVE
DEVICE #     DEVICE NAME
--------     -----------
 1           /dev/rst0
 2           /dev/rst1

Enter one of the device numbers (1-2) or (C)ancel : 1
```

You select the device by entering its number in the list. For example, you can enter 1 to select device **/dev/rst0**.

ON-Archive subsequently prompts you to mount a volume, as shown in the following example:

```
Please mount a volume on /dev/rst0 and press < RETURN > ** WRITE **
```

This message gives you time to mount a volume and also verifies your choice of a physical device.

See "Which Device Does onautovop Use?" on page 4-53 for an explanation of how **onautovop** selects a device.

How to Assign Logical and Physical Devices

You can use DEVICE parameters to associate multiple physical devices with a single logical device or to associate a single physical device with multiple logical devices.

You might assign multiple physical devices to a single logical device to group physical devices by type. For example, you might assign each of the following groups of physical devices to a separate logical device:

- Fast or slow tape (or disk) drives
- High-capacity or low-capacity storage units (9-track or 8-mm)
- Removable disk drives

You can also assign a physical device to *multiple* logical devices. This enables you to use the same device for more than one purpose.

For example, when you operate a very accessible high-speed tape device and a high-speed tape device in a secured room, you could create two logical devices for these devices and assign the secure device to both of them. When you archive payroll data you can select the secure device to ensure that the payroll data outputs at high speed and on a secure device. When you wish to archive other data on a high-speed tape device, you can select any available high-speed tape device, including the one in the secure area.

Using Remote Devices

ON-Archive allows you to use a remote tape device—that is, a tape device attached to another host computer. The remote device must be a *rewind device*; you cannot use nonrewind devices as remote devices.

You specify a remote tape device by placing *hostname:* in front of the device pathname. For example, the following DEVICE parameter specifies remote tape devices on the host **lockbox**:

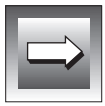
```
DEVICE secure   = lockbox:/dev/rst0, lockbox:/dev/rst1
```

Remote disk devices must be NFS-mounted; you do not need a hostname because the disk devices appear to be local instead of remote.

Requests that use remote devices must specify that the save sets be created on separate volumes (by using the APART qualifier, as explained in "Keeping Archives on Separate Volumes" on page 5-17).

ENGLISH

The first parameter in the **config.arc** file defines the default language used by ON-Archive. ON-Archive provides language files for English and French. For questions regarding the use of another language, contact your Informix representative.



Important: *The language parameter only changes the default language for* ***onarchive****. The* ***oncatlgr*** *program prints text as defined by the CATALOG MESSAGE parameter.*

To change the default language to French, replace the word ENGLISH with the word FRENCH on the language parameter.

ENGLISH

ERROR

The ERROR parameter specifies the location of error-message files. ON-Archive supplies the following two error-message files: **errmsg_E.dat** (English version) and **errmsg_F.dat** (French version).

language specifies the language: English or French.

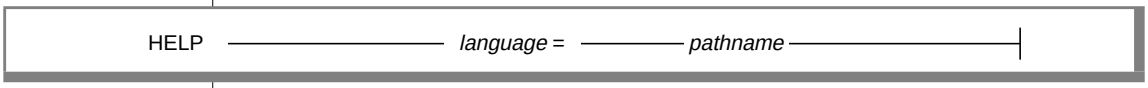
pathname is relative to **$INFORMIXDIR**. The *pathname* must begin with a slash (/). The *pathname* specifies the location and name of the error-message file.

The following example illustrates the ERROR parameter in English and French:

```
ERROR ENGLISH   = /arc_fls/ee.dat
ERROR FRENCH    = /arc_fls/ef.cat
```

HELP

The HELP parameter specifies the location of Help pointer files. ON-Archive supplies the following Help files: **hlp_E.hpf** (English) and **hlp_F.hpf** (French):



language specifies the language: English or French.

pathname is relative to **$INFORMIXDIR**. The *pathname* must begin with a slash (/). The *pathname* specifies the location and name of the Help file.

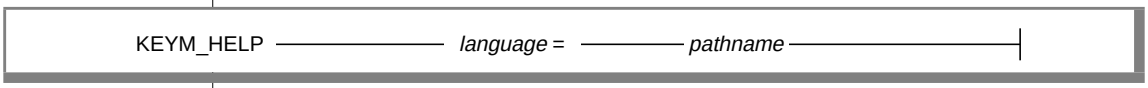
ON-Archive supplies Help files for **ondatartr** in the **$INFORMIXDIR/msg** directory. You cannot change the location of these files. The **ondatartr** help files are **hlp_r_E.hpf** (English) and **hlp_r_F.hpf** (French).

The following example illustrates the HELP parameter for English and French:

```
HELP ENGLISH    = /arc_fls/he.hpf
HELP FRENCH     = /arc_fls/hf.hpf
```

KEYM_HELP

The KEYM_HELP parameter specifies the location of Help files used by the **onkeymgr** utility. ON-Archive supplies the following help files for **onkeymgr**: **hlp_km_E.hpf** (English) and **hlp_km_F.hpf** (French).



language specifies the language, English or French.

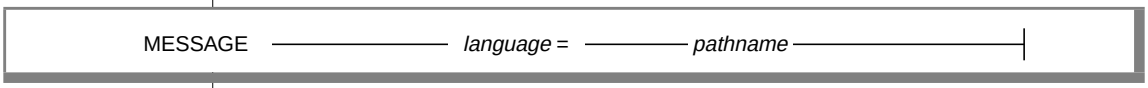
pathname is relative to **$INFORMIXDIR**. The *pathname* must begin with a slash (/). The *pathname* specifies the location and name of the help file for the **onkeymgr** utility.

The following example illustrates the KEYM_HELP parameter for English and French:

```
KEYM_HELP ENGLISH = /arc_fls/ke.hpf
KEYM_HELP FRENCH = /arc_fls/kf.hpf
```

MESSAGE

The MESSAGE parameter specifies the location of the message-format files. ON-Archive supplies the following message-format files: **fmt_E.dat** (English version) and **fmt_F.dat** (French version).



language specifies the language: English or French.

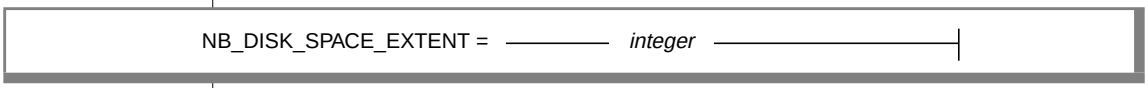
pathname is relative to **$INFORMIXDIR**. The *pathname* must begin with a slash (/). The *pathname* specifies the location and name of the message-format file.

The following example illustrates the MESSAGE parameter:

```
MESSAGE ENGLISH = /arc_fls/fe.dat
MESSAGE FRENCH = /arc_fls/ff.dat
```

NB_DISK_SPACE_EXTENT

The NB_DISK_SPACE_EXTENT parameter defines the size of the space allocated to each archive, copy or backup operation that uses a disk volume. When an archive, backup or copy operation fills the allocated space, it receives another allocation of the specified size. Allocating disk space in large blocks eliminates the need to continually check how much space remains on the volume when multiple operations use the same volume (disk directory). (See "Disk-Space Allocation During Concurrent Operations" on page 2-22 for more information.)



integer specifies a value in the range of 1 to 100 that, when multiplied by the value of the BLOCKSIZE qualifier, determines how much space you allocate on a disk volume each time a request needs more space.

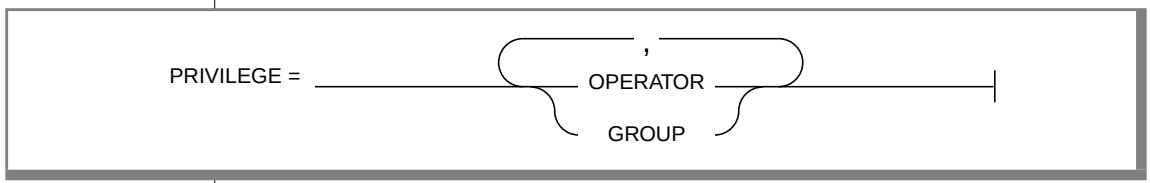
The following example defines the NB_DISK_SPACE_EXTENT parameter:

```
NB_DISK_SPACE_EXTENT = 30
```

Tip: *You can improve the performance of archives and backups that use disk volumes by using a large value for NBDISK_SPACE_EXTENT. You achieve the best performance by using the highest value allowed; typically 100. When you define the disk volume, you must specify a MAX_SPACE value large enough to hold at least one extent. When you perform concurrent operations to the same vset and volume, each volume must be able to hold as many extents as the number of possible concurrent operations.*

PRIVILEGE

The PRIVILEGE parameter specifies the privilege mode for ON-Archive. When you define the privilege mode, you can limit access to ON-Archive, as explained in "Privilege Modes" on page 2-25.



OPERATOR is the default value for the PRIVILEGE parameter. Operators are always the accounts called **root** and **informix**. They can access all ON-Archive commands, and possess the authority to perform all archive-and-retrieve operations, and can initialize all storage media for use by ON-Archive.

GROUP specifies that all users in the **super_archive** group can perform archive and retrieve operations, as long as they retain read access to the data and write access to the archiving media. (See "Creating a super_archive Group" on page 3-19 for information on the **super_archive** group.)

See the section that describes the command to see how each individual command behaves in the different privilege modes.

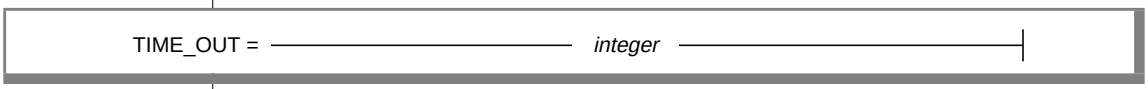
In the following example, ON-Archive grants access to **root**, user **informix**, and members of the **super_archive** group.

```
PRIVILEGE = OPERATOR, GROUP
```

Important: *Like all parameters, each time the PRIVILEGE parameter gets modified, you must stop and restart* ***oncatlgr*** *for the change to take effect. (See* *"The oncatlgr Utility" on page 9-11**.)*

TIME_OUT

The TIME_OUT parameter specifies the maximum number of minutes that ON-Archive waits for an operator to mount a tape or disk volume before it terminates with a time-out condition during attended operations.



integer specifies the number of minutes before a time-out condition.

The following TIME_OUT specification causes ON-Archive to wait 15 minutes for an operator to mount a volume on a physical device. If the mount remains incomplete after 15 minutes, ON-Archive terminates.

```
TIME_OUT = 15
```

Creating a super_archive Group

OPERATOR is the default privilege mode of ON-Archive, and only **root** and **informix** can execute ON-Archive requests and initialize media. When you want to allow specific users to execute archive-and-retrieve requests without requiring them to know the **root** password, you must take the following actions:

- Change (or add) the privilege mode to GROUP. (See "PRIVILEGE" on page 3-17.)
- Place the users that need capabilities similar to **root** and **informix** in an operating-system group called **super_archive**.

To create the **super_archive** group, you must edit the **/etc/group** UNIX file, add a new group named either **super_archive** or **superarc**, and put the user names that you want to execute ON-Archive requests on the group list. The new line in the group file must look something like the following example:

```
super_archive:*:50:username[,username]
```

where:

super_archive	is the new group name. It *must* be called either **super_archive** or **superarc**.
50	is the group id, that must be unique within the **/etc/group** file.
username [,*username*]	is the list of users who belong to the **super_archive** group, and can therefore execute archive or retrieve requests.

See your operating-system documentation for more complete instructions on creating a group and adding new members to a group.

Chapter

4

Using ON-Archive

This chapter explains how to use ON-Archive to perform the following tasks:

- Start and stop the cataloger and the **onarchive** programs
- Use the command interface
- Use the menu interface
- Create and execute requests
- Define volume sets (vsets) and volumes
- Use **ondatartr** in emergency situations
- Use unattended operations

Starting and Stopping the Cataloger

You must start the **oncatlgr** program, also known as *the cataloger*, before you start either attended or unattended operations. You can find reference information about **oncatlgr** in "The oncatlgr Utility" on page 9-11.

Starting the Cataloger

Universal Server must operate in on-line mode to start **oncatlgr**. As **informix** or **root**, execute the following command at the operating-system prompt to start **oncatlgr**:

```
% start_oncatlgr
```

You must make sure that only one cataloger runs for each instance of Universal Server. When you start the cataloger, **start_oncatlgr** first checks for the existence of any other **oncatlgr** processes. When another **oncatlgr** process is running, **start_oncatlgr** asks if you want to stop running the cataloger.

See "The start_oncatlgr Script" on page 9-32 for more information on starting **oncatlgr**.

Which oncatlgr for Which Universal Server?

When multiple Universal Server instances reside on one computer, you can determine which **oncatlgr** process is associated with a particular instance of Universal Server by examining the file **/tmp/oncatlgr.out.*pid***, where *pid* is the process id of the **oncatlgr** process. This file contains the server number of the Universal Server instance with which the particular **oncatlgr** process is associated. The following operating-system command, for example, readily identifies the process id and **/tmp/oncatlgr.out.*pid*** file that is associated with a particular instance of Universal Server:

```
% grep Server /tmp/oncatlgr.out*
```

Starting the Cataloger Automatically

Because you must run **oncatlgr** before you can enter or execute any ON-Archive commands, you might want to ensure that it is running whenever Universal Server is running. To start **oncatlgr** automatically when you start up Universal Server, add the following command to your start-up command file.

```
$INFORMIXDIR/bin/start_oncatlgr
```

This command *must not* precede the start-up of Universal Server.

Stopping the Cataloger

You must stop the cataloger before you bring Universal Server off-line to avoid futile attempts to communicate with Universal Server and spurious error messages.

To stop the **oncatlgr** process, execute the following command as **informix** or **root** at the operating-system prompt:

```
% stop_oncatlgr
```

When more than one instance of Universal Server is running on the computer, you can make multiple **oncatlgr** processes run. A prompt asks you if you want to kill each **oncatlgr** process that is running. You must take care to stop only the **oncatlgr** associated with the instance of Universal Server you are using. See "The stop_oncatlgr Script" on page 9-34 for information on the command to stop **oncatlgr**.

Starting onarchive

Once you start the cataloger, you can start the **onarchive** utility, which enables you to enter ON-Archive commands and requests. The **onarchive** program provides both a command-line interface and a menu interface. You can use either one to enter ON-Archive commands or requests.

No special privilege is required to run **onarchive**, although commands behave differently depending on the ON-Archive privilege mode. (See "Privilege Modes" on page 2-25 for a description of privilege modes.)

Enter the following command at the operating-system prompt to start **onarchive** with the command-line interface:

```
% onarchive
```

See "Using the ON-Archive Menu Interface" on page 4-17 for information on how to select the menu interface when you start **onarchive**. That section also describes how to use the MENU command to switch from the ON-Archive command line to the menu interface.

Using ON-Archive Commands

The following list shows the organization of ON-Archive command descriptions in this manual:

- This chapter describes the DEFINE / VSET, DEFINE / VOLUME, EXECUTE, EXIT, HELP, and MENU commands.
- Chapter 5, "Creating an Archive," describes the ARCHIVE command. Chapter 6, "Backing Up the Logical Log," describes the BACKUP command.

- Chapter 7, "Restoring Universal Data," describes the RETRIEVE / DBSPACESET and RETRIEVE / LOGFILE commands.
- Chapter 8, "Administering ON-Archive," describes the remainder of the ON-Archive commands.

The following table lists and briefly describes the ON-Archive commands:

Command	Purpose	Page
ARCHIVE	Creates a full-system archive, or an archive of specified dbspace sets.	5-12
BACKUP / LOGFILE	Backs up logical log files.	6-9
CANCEL	Changes the status of specified requests to CANCELLED.	8-4
CATALOG	Reads the vset or volume and stores information concerning it in the catalog.	8-21
COPY / VSET	Copies the specified vset or volume to another vset.	8-24
DEFINE / DBSPACESET	Defines a dbspace set for the specified dbspaces.	5-5
DEFINE / VSET	Defines a vset.	4-31
DEFINE / VOLUME	Defines a new volume for the specified vset.	4-38
DELETE / DBSPACESET	Deletes a dbspace-set definition from the catalog.	8-48
DELETE / USER	Removes a user or users from the access list for a vset.	8-30
DELETE VOLUME	Deletes a volume definition from the catalog.	8-31
DELETE / VSET	Deletes a vset from the catalog.	8-32
EXECUTE	Executes a specific request, or all the requests for a vset.	4-27
EXECUTE / VSET	Executes requests that contain the same vset destination.	8-33
EXIT	Terminates ON-Archive.	4-17

(1 of 3)

Command	Purpose	Page
HELP	Gets help on an ON-Archive topic.	4-23
LIST / DATABASE	Shows the table names for a database, the dbspaces and blobspaces in which the database tables reside, and the dbspace set in which the dbspaces and blobspaces reside.	8-49
LIST / DBSPACESET	Displays information for either a specific dbspace set or for all dbspace sets.	8-52
LIST / DEFAULT	Displays all current default settings for the ON-Archive qualifiers.	8-68
LIST / LOGFILE	Displays information about backed-up logical-log files.	8-56
LIST / LOGRECORDS	Displays the contents of a backed-up logical-log file.	8-58
LIST / METHOD	Displays the names and descriptions of the compression and encryption methods available to ON-Archive.	8-66
LIST / REQUEST	Displays information on all past and current requests in the catalog.	8-8
LIST / VOLUME	Displays some or all of the volumes of a vset.	8-36
LIST / VSET	Displays information about vsets in the catalog.	8-38
MENU	Uses the menu interface.	4-18
MODIFY / COMMAND	Modifies a request stored in the catalog.	8-11
MODIFY / DBLOGGING	Marks the specified databases to change their logging status.	8-62
MODIFY / VOLUME	Changes a volume definition.	8-42
MODIFY / VSET	Changes a vset definition.	8-44
REMOVE / REQUEST	Removes a specified request from the catalog.	8-17

(2 of 3)

Command	Purpose	Page
REMOVE / FAILED_REQUEST	Removes all requests with a status of FAILED from the catalog.	8-15
RETRIEVE / DBSPACESET	Restores the specified dbspaces from an archive.	7-14
RETRIEVE / LOGFILE	Starts a warm logical restore operation.	7-19

(3 of 3)

Command and Qualifier Syntax

An ON-Archive command is composed of a command *keyword*, followed by a number of *qualifiers* separated by slash characters (/). The keyword specifies the action to be performed, and the qualifiers specify various aspects of the action (the vset and volume to use, and so on). For example, the following command lists the contents of the second volume in the vset named **accting**:

```
LIST/VOLUME=2/VSET=accting
```

Many commands perform dramatically different actions based on the qualifiers you use with them. For example, each of the following qualifiers creates a distinctly different MODIFY command with a completely different purpose:

- MODIFY / COMMAND
- MODIFY / DBLOGGING
- MODIFY / VOLUME
- MODIFY / VSET

This section outlines the rules about qualifiers, qualifier parameters, qualifier ordering, qualifier overriding, capitalization, and abbreviation. "Groups of Qualifiers" on page 8-67 provides syntax diagrams and descriptions for the following groups of qualifiers:

- "Archive and Backup Qualifiers" on page 8-70
- "Before and Since Qualifiers" on page 8-75
- "Output Qualifiers" on page 8-77
- "Compression and Encryption Qualifiers" on page 8-78
- "Decompression and Decryption Qualifiers" on page 8-80

Command Qualifiers

A qualifier specification always starts with a slash (/), followed by a qualifier name. Depending on the qualifier, ON-Archive could require parameters. For example, the VSET qualifier requires a parameter, which is the name of the vset:

```
/VSET=myvset
```

Spaces are allowed before the slash (/) and on each side of the equal sign (=), but not between the slash and the qualifier.

Some options accept the keyword NO in front of the qualifier name to specify negation. For example, the APART qualifier specifies that you must keep a save set apart (on different media) from other save sets. NOAPART specifies that you can keep a save set on a volume with other save sets.

Qualifiers with Parameters

Depending on the qualifier, parameters are either compulsory, optional, or not accepted. You must always separate the parameters from the qualifier name by an equal sign (=). When you must specify more than one value as a parameter, you must separate the different values using commas (,) and you must enclose the whole list in parentheses "( )". When you use a negation qualifier, such as NOLOG, no parameter is expected.

Qualifier Ordering

This manual illustrates the syntax of each ON-Archive command with a syntax diagram. These syntax diagrams often impose an order on the qualifiers to indicate which options are mandatory and whether you must use them with one another. However, the order in which the qualifiers appear in a command has no effect on the interpretation of the command.

For example, the following two commands are equivalent:

```
ARCHIVE/DBSPACESET=*/NOTIFY

ARCHIVE/NOTIFY/DBSPACESET=*
```

Qualifier Overriding

The same qualifier can appear several times in a command, but ON-Archive uses only the last one. For example, in the following LIST/FILE command the second occurrence of the DBSPACESET qualifier overrides the first one:

```
LIST/DBSPACESET=(*)/BEFORE=01-APR-1993/DBSPACESET=root
```

The result is the same as if the command was entered as shown in the following command line:

```
LIST/DBSPACESET=root/BEFORE=01-APR-1993
```

The qualifiers within each of the following sets can override each other. They are said to be in the same *override set*:

- BRIEF and FULL
- ONSITE and OFFSITE and ANYWHERE

When you use multiple qualifiers from the same override set, ON-Archive uses only the one that appears last.

Capitalization and Spaces

You can enter ON-Archive commands and qualifiers in UPPERCASE characters, lowercase characters, or any combination of the two. The ON-Archive command-line interpreter automatically converts all entries (except for parameters) into uppercase characters.

Names of objects such as vset names, dbspace-set names, and so on are case sensitive.

You can use multiple spaces and/or tabs where a single space or tab is legal. The command interpreter discards excess spaces.

Abbreviation of Commands and Qualifiers

You can abbreviate all ON-Archive commands and qualifiers to their shortest unique length. For example, you can abbreviate the command EXECUTE to EXE, but EX is not accepted because it is not clear whether it indicates EXECUTE or EXIT.

Qualifier Default Values

To reduce command length, ON-Archive uses default values for many commonly used command qualifiers. When a qualifier has a default value, you do not need to specify the qualifier to invoke the default value.

The ON-Archive qualifier defaults specify the following values:

- Default block size to use when writing to media
- Default compression routine used for requested data compression
- Default number of backup copies created when storing data

You can modify the default values for ON-Archive qualifiers on a system-wide or per-user basis.

Where Qualifier Default Values Are Specified

You can specify default values for ON-Archive qualifiers in the following places:

Internal defaults

ON-Archive sets internal defaults. Defaults specified in the original **$INFORMIXDIR/etc/oper_deflt.arc** file reflect the values used for the internal defaults.

The language parameter (ENGLISH) in the ON-Archive configuration file, **config.arc**, defines the default language. No internal default exists for the language parameter. Figure 4-1 on page 4-14 shows the contents of the default **oper_deflt.arc** file, which is found in the directory **$INFORMIXDIR/etc**.

Operator defaults

The operator defaults are read from the file in **$INFORMIXDIR/etc** specified by the DEFAULT parameter in the ON-Archive configuration file, **config.arc**. The default filename is **oper_deflt.arc**. You can set the **ARC_DEFAULT** environment variable to specify the name of the file that contains your operator default values. The file must reside in the directory **$INFORMIXDIR/etc**.

Personal defaults

Personal defaults, when defined, are read from a file created by the user and specified by their **ARC_DEFAULT** environment variable.

```
! This file is used only if users haven't defined their own default file.


! Starting language (override the configuration language).
ENGLISH

! These defaults are the same as the system's defaults.
! Modified as needed.
/NOAPART
/BLOCKSIZE=65536
/BRIEF
/NOCOMPRESS
/NODECOMPRESS
/COPIES=1
/NODECRYPT
/DENSITY=0
/NOEXPIRY_DATE
/NOENCRYPT
/LEVEL=0
/NOLOG
/MAX_SPACE=0
/NONOTIFY
/PROTECTION=RWD
/NOTRANSIT
/NOVERIFY
```

Figure 4-1
The oper_deflt.arc File

Which Defaults Are Used?

ON-Archive uses the following order of precedence in selecting default specifications:

1. When a personal default file is defined, the specified values of the user qualifiers override any operator or ON-Archive internal default values for that user. Values unspecified in the personal default file use the internal defaults.
2. When no personal default file is defined, the operator default file is used. The values of those qualifiers override any ON-Archive internal defaults and become the default for all users that do not have a personal, user-specific default parameter file.
3. When a parameter is neither specified in a user or operator default file, the ON-Archive internal defaults apply.

To modify the default values for qualifiers on a system-wide basis, you can take either of the following actions:

- Edit the existing **oper_deflt.arc** file and make the desired changes.
- Create an alternative operator default file in **$INFORMIXDIR/etc** and modify the DEFAULT configuration parameter in the **config.arc** file to specify that filename.

For example, when the internal default value is BRIEF, and the operator default is FULL, the following table illustrates what different users experience as a default:

User	What They See
User without a personal file	operator default of FULL
User with a personal file that specifies BRIEF	BRIEF
User with a personal default file, but without BRIEF or FULL defined	internal default of BRIEF

Status of a Failed Command

When an error occurs during the execution of a command, or during the start-up or shutdown of the **onarchive** process, ON-Archive returns a status of 1 (one) to the operating system. The status enables you to write scripts like the following UNIX shell script, that checks the status after each command. The script defines a volume, creates an archive request, and starts **onautovop**.

```
onarchive "define/vol/vset=vset1/virtual=(/vset1/V4)/max_space=9999" > outfile
if [ $? -ne 0 ] ; then
    echo onarchive failed while defining /vset1/V4 | mail DBA
    exit 5
fi
onarchive "archive/dbspaceset=*/vset=vset1/level=2" >> outfile
if [ $? -ne 0 ] ; then
    echo onarchive failed to create the archive request | mail DBA
    exit 6
fi
onautovop >> outfile
if [ $? -ne 0 ] ; then
    echo onautovop failed | mail DBA
    exit 7
fi
exit 0
```

Using the ON-Archive Command-Line Interface

The following greeting message appears when you start **onarchive** with the command-line interface. (Text might differ slightly on different computer systems.)

```
% onarchive
ON-Archive:  Version 7.1
Copyright (C) 1994 - Informix Software, Inc.
Onarchive>
```

The prompt `Onarchive>` indicates the command line where you can enter ON-Archive commands.

Entering a Command

Enter a command to the right of the `Onarchive>` prompt and press the RETURN key to terminate the command. The following example illustrates a simple ARCHIVE command:

```
Onarchive>ARCHIVE/DBSPACESET=*
```

ON-Archive processes the command and then displays a new `Onarchive>` prompt to indicate it is ready for another command.

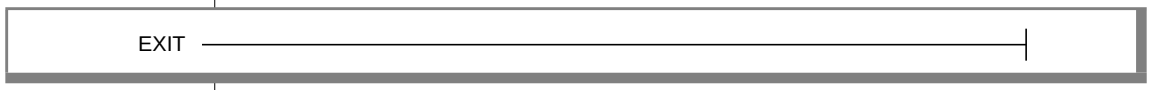
If you need to continue a command on another line because it is too long to fit on one line, enter a dash (-) at the end of the line. The following prompt indicates that ON-Archive is expecting the continuation of the command. (Note the initial underscore character.)

```
_Onarchive>
```

The EXIT Command

The EXIT command terminates ON-Archive.

The EXIT command has no restrictions.



EXIT Example

The EXIT command in the following example terminates ON-Archive:

```
Onarchive> exit
```

Using the ON-Archive Menu Interface

When you use the menu interface, you do not need to memorize the ON-Archive command syntax. Instead, every command is presented as a form with fields where you enter information. Each field represents a qualifier or a qualifier parameter.

You can invoke the menu interface in two ways. The first method is to enter the word `menu` following the **onarchive** command when you start **onarchive** from the *operating-system* command line, as shown in the following example:

```
% onarchive menu
```

The second method is to enter the MENU command on the ON-Archive command line, as explained in the next section.

The MENU Command

The MENU command calls the menu interface from the command-line interface. You cannot call this command in batch mode because you can only use the menu interface interactively.

The MENU command has no restrictions and no qualifiers.

MENU Example

Enter the MENU command from the ON-Archive prompt as shown in the following command line:

```
Onarchive> MENU
```

Using the ON-Archive Main Menu

The ON-Archive main menu appears as shown in Figure 4-2.

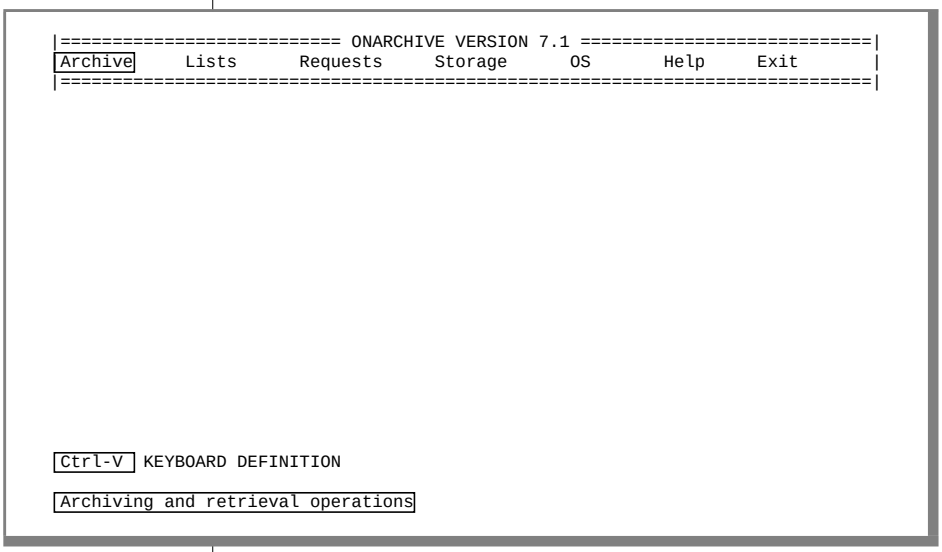


Figure 4-2
The ON-Archive Menu Interface

The main menu is divided into the following sections:

- The top section displays the choices available to you.
- Use the key combination on the bottom to display the current keyboard definition.
- The status line on the bottom describes the option that the cursor is highlighting.

To exit the menu interface, select `Exit` from the main menu, and `Command interface [1]` from the submenu.

See "Getting Help" on page 4-23 for information on getting help specifically for the menu interface.

Using ON-Archive Menus

The menu interface has *pull-down, pop-up* style menus. When you select an option, a submenu appears to pull down from the menu item selected. When you complete the option, the window pops up.

The menu interface also provides *window*s. When you select a submenu or screen, ON-Archive places it on top of the previous screen. The effect is the same as placing one piece of paper on top of another on a desk. When you move the top window you can access the one underneath it. The information in a window remains intact even when another window covers it.

You can find all the command-line interface commands in the menu interface. The following sections discuss the various methods of selecting an option from the menu interface.

Selecting Items from the Main Menu

You can select an option from the main menu in the following ways:

- Press the first letter of an option on the main menu to make the submenu for that option appear.
- Move the cursor between menu options with the *cursor-up* and *cursor-down* keys, as explained in "Using Keyboard Commands" on page 4-20. When you move the cursor to the option you want, press the RETURN key to select it.

Selecting Items from Submenus

You can select an option from a submenu as shown in the following list:

- Enter the numeric value shown in square braces, such as [1], immediately following each submenu option.
- Enter the first letter of a submenu option to move the cursor to that option. When multiple lines begin with the same letter, enter the letter the number of times necessary to place the cursor on the option you want. When the cursor resides on the correct line, press the RETURN key to select the option.
- Move the cursor to the appropriate line using the *cursor-up* and *cursor-down* keys (as described in "Using Keyboard Commands"). When you move the cursor to the option you want, press the RETURN key to select it.

Exiting a Menu

To exit from any menu item and return to the next higher menu, press CTRL-C.

To move from one submenu to the submenu of a main-menu option on the left or the right of the current one, press the *Cursor-Left* or *Cursor-Right* keys. The effect of moving from one submenu to another in this manner is called *moving up and over.*

Using Keyboard Commands

When you press CTRL-V, ON-Archive presents the following list of the keyboard commands that you can use:

Function	Keys
Accept	ESC
Cancel	CTRL-C
Help	CTRL-W
Display errors	CTRL-E

(1 of 2)

Function	Keys
Refresh screen	CTRL-R
Insert / replace	CTRL-A
Clear field	CTRL-U
Delete to end of field	CTRL-D
Delete character	CTRL-X
Backspace	CTRL-H
Cursor left	LEFT ARROW
Cursor right	RIGHT ARROW
Cursor up	UP ARROW
Cursor down	DOWN ARROW

(2 of 2)

To change these keyboard definitions, create a new **tctermcap** file and setting the environment variable **ARC_KEYPAD** to point to the new file. Instructions for creating a new **tctermcap** file reside in the default file **$INFORMIXDIR/etc/tctermcap**.

Switching Input Modes

Press CTRL-A to switch between insert and typeover mode.

Validation of Input

ON-Archive validates input as much as possible at the field level. ON-Archive does not permit the cursor to leave a field until you enter the data in the correct format. You can detect the field-level validation checks for format and content errors without accessing the ON-Archive catalog. For example, asking for an archive request to be started in the past could result in a field-level error but specifying an invalid vset could not.

Error Messages

ON-Archive displays all error messages in an error window on the bottom of your screen. You must press a key to acknowledge an error message and clear it from the window. When you generate multiple error messages,ON-Archive displays them sequentially in the error window. The bottom right-hand corner displays a counter in the format *x of y* to indicate the number of the current error message relative to the total number of error messages. For example, the counter could display the following line for the first of three error messages:

```
1 of 3
```

For a list of the latest command errors, press CTRL-E. This screen displays all the ON-Archive error messages that the last command generated. When no errors were detected in the current session, the list remains empty. The list reinitializes only when a new error is detected.

Executing Operating-System Commands

You can execute operating system commands from the menu interface. This feature is not available from the command-line interface.

Select the OS option on the main menu to gain access to the operating system.

Exiting the Menu Interface

To quit the menu interface, select the EXIT option from the main menu. The EXIT option submenu pulls down and displays two options: Command Interface and Operating System. Select the first option when you wish to switch to the ON-Archive command interface. Select the second option when you wish to exit ON-Archive.

Getting Help

Help is available for all ON-Archive commands and terms through the HELP command described in the following section.

The HELP Command

The HELP command initiates an interactive Help session to display information about an ON-Archive command or topic. The Help session first lists the topics available and presents you with a prompt that looks like the following example:

```
Topic?
```

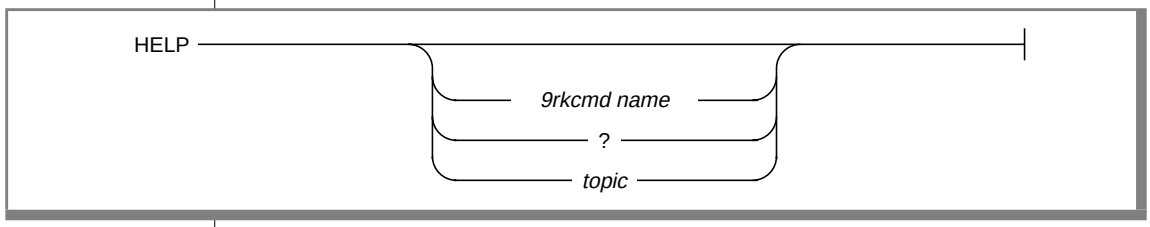
In response to the `Topic?` prompt, you can take the following actions:

- Type the name of the command or topic and press RETURN.
- Type a question mark (?) and press RETURN to display all the topics available for the current level.
- Press RETURN one or more times to exit from Help.

You can abbreviate topic names, although ambiguous abbreviations can result in all matches being displayed.

When you initiate a Help session, you can type the name of the topic and subtopic to specify a subtopic directly.

The HELP command has no restrictions.



cmd name	specifies the name of an ON-Archive command. Like all ON-Archive commands, you can abbreviate it.
?	displays a list of the Help topics. You can see a list of the Help topics by using the ?, either with the HELP command or at the `Topic?` prompt once you start a Help session.
topic	specifies the name of a Help topic.

HELP Example

The following example illustrates how to use the HELP command to get information on the RETRIEVE command:

```
Onarchive> HELP RETRIEVE

RETRIEVE
Help text on RETRIEVE command ...
Additional information available:
            list of sub-topics ...
RETRIEVE sub-topic?
```

To get help on any of the RETRIEVE command subtopics, enter the subtopic at the `sub-topic?` prompt.

Exiting Help

To quit Help, keep pressing the RETURN key until the `Onarchive>` command-line prompt appears.

Getting Help from the Menu Interface

When you select the Help option from the main menu, you can obtain Help in the menu interface for ON-Archive commands and terms. This option gives you the same information that the HELP command provides.

You can also get Help specifically for the menu interface. You can press CTRL-W at any time to receive context-sensitive Help for the menu interface. The Help information corresponds to the field where the cursor currently resides.

Creating and Executing Requests

Requests are commands that do not execute immediately. Instead, the command generates a request and stores it in the ON-Archive catalog. You can execute requests either explicitly using the EXECUTE command or implicitly using the virtual operator, **onautovop**. See "ON-Archive Commands and Requests" on page 2-5 for a complete description of commands and requests. See "Using Unattended Operations" on page 4-50 for information on how **onautovop** executes commands.

Creating a Request

You enter an ON-Archive request in the same way that you enter other ON-Archive commands. The following ON-Archive commands become requests:

- ARCHIVE
- BACKUP
- COPY
- RETRIEVE / DBSPACESET
- RETRIEVE / LOGFILE
- REMOVE with WAIT

See "ON-Archive Commands and Requests" on page 2-5 for a description of a request. When ON-Archive stores a request in the catalog, it assigns the request a unique *request id* (rid). For example, when you enter a command like the following ARCHIVE command:

```
Onarchive> ARCHIVE/DBSPACESET=cust_set/LEVEL=0/VSET=cust
```

ON-Archive responds with a message like the one shown in the following line, to display the request id:

```
Request 00000015 registered in the catalog
```

The owner of a request is the user who creates the request. The owner is not necessarily the same person as the user who executes the request.

You can execute a request in the following two ways:

- Manual execution
- Automatic execution

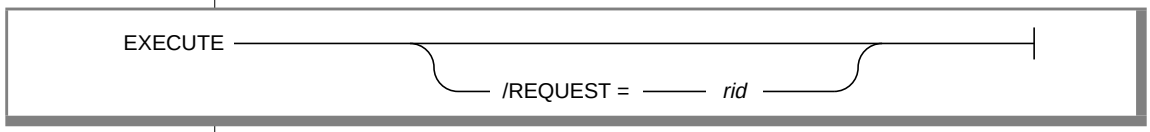
When you submit an EXECUTE command, you can execute a request manually. You can also make the **onautovop** process execute requests automatically. See "Using Unattended Operations" on page 4-50 for information on how **onautovop** executes requests.

The EXECUTE Command

The EXECUTE command enables you either to execute a specific request or, when you do not specify a request, to review each unprocessed request and optionally execute or cancel it.

The following restrictions apply to the EXECUTE command:

- When ON-Archive is running in OPERATOR privilege mode, the user must log in as **informix** or **root**. These users can execute requests created by any users.
- When ON-Archive is running in GROUP privilege mode, the user must log in as part of the **super_archive** group. Users in the **super_archive** group can execute requests made by any other user in the **super_archive** group.
- Users must create appropriate privileges on the vset and volume the request is writing to or reading from for the request to execute successfully.



REQUEST ON-Archive starts the execution of the specified request. After the request is completed, ON-Archive returns to the operating-system prompt where the EXECUTE command was issued.

rid specifies the request id of a request currently in the catalog to execute. Use "The LIST / REQUEST Command" on page 8-8 to see available requests.

When the REQUEST qualifier is omitted, interactive mode starts and the EXECUTE command presents all requests for possible execution.

See "The EXECUTE / VSET Command" on page 8-33 for information on how to execute all the requests for a specific vset.

The LIST / REQUEST command allows you to examine requests that were created before or after a particular date, that are owned by a specific user, or that have a particular status. See "The LIST / REQUEST Command" on page 8-8 to see how to display information about the requests currently stored in the catalog. See "The CANCEL Command" on page 8-4 for information on cancelling a request. See "The REMOVE / REQUEST Command" on page 8-17 and "The REMOVE / FAILED_REQUEST Command" on page 8-15 for information on removing requests. See the "The MODIFY / COMMAND Command" on page 8-11 for information on how to modify a request.

EXECUTE Examples

The EXECUTE command in the following example executes request id 4:

```
Onarchive> EXECUTE/REQUEST=4
```

The EXECUTE command in the following example starts interactive mode:

```
Onarchive> EXECUTE

Type the (number) to execute a request, or e(X)it to leave the
execute mode.
Execute>
```

Executing Requests Automatically

You can use the **onautovop** program to execute requests automatically. See "Using Unattended Operations" on page 4-50 for information on how to execute requests automatically. See "The onautovop Utility" on page 9-9 for a description of **onautovop.**

Repeating Requests

You can also submit a request that can be repeated. For example, you can request that ON-Archive performs an archive at midnight on a particular day and repeat that request repeated each week, thus automating your archive schedule. See "Wait and Repeat Qualifiers" on page 8-82 for details on how to repeat requests.

Request Statuses

A request always has one of the following statuses:

- CANCELLED
- COMPLETED
- EXECUTING
- FAILED
- NEW
- PARENT
- UNCOMPLETED

During its lifetime, the status of a request changes several times. When an ARCHIVE, BACKUP, RETRIEVE, or COPY command is submitted, ON-Archive initially sets its status to NEW. When the request executes, its status changes to EXECUTING. When the request is completes, its status changes to COMPLETED when the execution succeeds or to FAILED when an error occurs during execution.

When a request specifies that ON-Archive archives multiple copies of the data, the status of the request changes to PARENT after the first copy is completed. A COPY request is automatically submitted to generate each of the subsequent copies. The COPY request is referred to as a *child-copy* request. Once all the child-copy requests are completed, the status of the original request changes to COMPLETED.

When a child-copy request has a status of CANCELLED or FAILED, the status of its parent changes to UNCOMPLETED.

Requests that specify and use transit media create a similar status pattern. (See "What Is a Transit Volume Set?" on page 2-11.)

Where Are Errors Sent?

When an error occurs when a request executes, ON-Archive reports the error in a manner that depends on how the request was executed. The following list describes the different ways that ON-Archive reports errors:

- When **onarchive** executes the request, ON-Archive mails the error message to the terminal.
- When **onautovop** executes the request, ON-Archive mails the error to the owner of the request.
- When the request includes the NOTIFY or LOG qualifier, regardless of whether **onarchive** or **onautovop** executes the request, ON-Archive writes the error to a file **arc*xxxxxxx*.log** in the current working directory where xxxxxxx is the request ID.

Using Volume Sets and Volumes

Before you can perform an archive or back up logical log files, you must define the vsets and volumes that ON-Archive uses to store the information. You use the DEFINE/VSET and DEFINE/VOLUME commands to create vsets and volumes. See "What Are Volume Sets and Volumes?" on page 2-9 for descriptions of these objects.

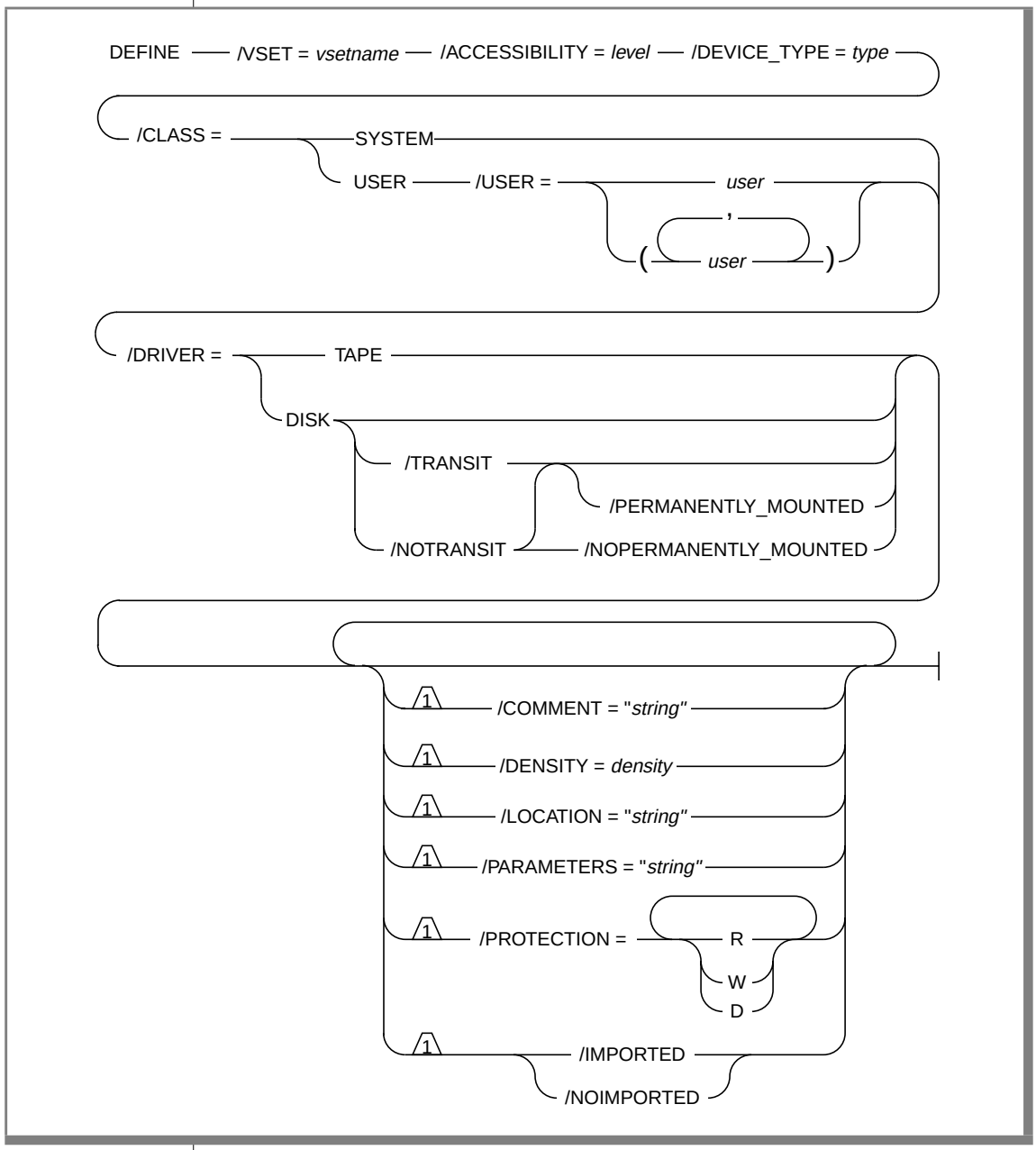
You must define a vset with the DEFINE/VSET command before you define any associated volumes.

The DEFINE/VSET Command

The DEFINE/VSET command defines a new vset for ON-Archive.

The following restrictions apply to the DEFINE/VSET command:

- When ON-Archive is running in OPERATOR privilege mode, only **informix** or **root** can define a vset.
- When ON-Archive is running in GROUP privilege mode, only users who are in the **super_archive** group can define a vset


DEFINE
/VSET = vsetname
/ACCESSIBILITY = level
/DEVICE_TYPE = type
/CLASS =
SYSTEM
USER
/USER =
user
,
(
user
)
/DRIVER =
TAPE
DISK
/TRANSIT
/PERMANENTLY_MOUNTED
/NOTRANSIT
/NOPERMANENTLY_MOUNTED
1
/COMMENT = "string"
1
/DENSITY = density
1
/LOCATION = "string"
1
/PARAMETERS = "string"
1
/PROTECTION =
R
W
D
1
/IMPORTED
/NOIMPORTED

VSET specifies the vset name.

vsetname is the name of the vset. You must use a unique name within an Universal Server database server and that name cannot exceed 17 characters in length. You can use letters, numbers, or symbols for the character symbols. You use the vset name in the ARCHIVE and BACKUP commands to specify the destination of the backup data.

ON-Archive assigns a unique internal numeric id to the vset. ON-Archive uses the numeric vset id internally; you do not use it in ON-Archive commands. ON-Archive stores the vset id in the ON-Archive catalog and applications can directly access the catalog.

ACCESSIBILITY defines the accessibility of the vset. When automatic vset selection occurs, ON-Archive is more likely to choose a highly-accessible vset than a less-accessible vset. Automatic selection occurs when an archive or backup request does not specify a vset. This qualifier has no default value. You can change vset accessibility at any time with the MODIFY/VSET command.

The accessibility of a vset indicates its priority relative to other vsets. For example, suppose you define three vsets: one for a high-capacity tape drive, one for a fast tape drive, and one for a disk directory. You might indicate your priorities for these devices by deciding that the vset that uses the high-capacity drive must be most accessible, the vset that uses the fast tape drive must be next, and the vset that uses the disk directory must be the least accessible.

level is an integer from 0 to 99, where 0 means the most accessible and 99 means the least accessible.

DEVICE_TYPE specifies the logical device (type of device) for the vset. The logical device must be defined on a DEVICE parameter in the ON-Archive configuration file. When you use the vset for an archive or backup, ON-Archive asks you to select one of the physical devices you listed for the logical device. You can change the DEVICE_TYPE at any time with the MODIFY/VSET command. This qualifier has no default value. See "How ON-Archive Organizes Data" on page 2-9 for an explanation of the relationship between logical and physical devices.

type specifies the name of a logical device.

CLASS Specifies whether all users can access the vset (SYSTEM class), or only a select group of users can access the vset (USER class). This qualifier has no default value. See "Access-Control Lists" on page 2-26 for background information on restricting access to a specific group of users.

SYSTEM All users can use the vset.

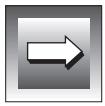
USER Only the users defined in the access-control list, specified by the USER qualifier, can access the vset. When you attempt to create a USER vset without any users on the access list you receive an error message and ON-Archive does not create an access list. You can add users to an existing access list using the MODIFY/VSET command.

USER specifies users in the vset access-control list. Only specified users can archive or retrieve data to or from the vset.

user is a valid operating-system user id.

DRIVER — specifies the device driver for the vset—that is, it specifies the characteristics of the logical device and, consequently, the type of access (direct or sequential) used for the vset. You can use any device that can emulate a disk or a tape device for archives and backups. This qualifier has no default value.

- TAPE — specifies that you must use a device driver for tape, or a device that emulates tape.
- DISK — specifies that you must use a device driver for disk, or a device that emulates disk.



Important: *Once defined, you cannot change the device driver for a vset.*

TRANSIT — specifies that the vset is a transit vset. ON-Archive uses a transit vset for temporary overflow storage space, when necessary, when archiving or backing up to a tape vset during unattended operations. You cannot direct data to a transit vset. ON-Archive writes to the transit vset transparently, when it needs it. A transit area or vset is not a final storage destination. It is a staging area. See "What Is a Transit Volume Set?" on page 2-11 for a description of a transit vset.

When you include the TRANSIT qualifier, ON-Archive automatically defines vset as permanently mounted. You must assign the value DISK to the DRIVER qualifier. Do not confuse this TRANSIT qualifier with the TRANSIT qualifier in ARCHIVE, BACKUP, RETRIEVE, and COPY commands. That TRANSIT qualifier specifies that you must use a previously defined transit vset, when necessary, during execution of the request.

NOTRANSIT — specifies that the vset is not a transit vset.

PERMANENTLY_MOUNTED — specifies that the vset is permanently mounted. The most common permanently-mounted vsets are disk vsets. The default is PERMANENTLY_MOUNTED when the vset is defined as a transit vset (see the TRANSIT qualifier) and is NOPERMANENTLY_MOUNTED for other vsets.

NOPERMANENTLY_MOUNTED specifies that the vset is not permanently mounted. The most common type of nonpermanently mounted vsets are tape vsets. NOPERMANENTLY_MOUNTED vsets indicate that operator intervention—to load a tape, for example—is required.

COMMENT specifies a descriptive comment about the vset. ON-Archive stores the comment in the catalog.

string is a string of text enclosed in double quotes. The string cannot exceed 80 characters in length.

DENSITY specifies the vset density in bits per inch (bpi). You must only use density specification when a device requires it. The default is 0.

density When 0, ON-Archive uses the default density of the device being used.

Important: *Once you define a vset, you cannot change the density definition.*

LOCATION a text string that identifies the physical location of the vset when it is not in use.

string is a text string. It cannot exceed 80 characters in length.

In addition to the location description, ON-Archive maintains a location status for each vset in the catalog. The possible statuses in the catalog are Y (onsite), N (off-site), or U (unknown). The status defaults to Y (onsite) when you create the vset. You can change the status using the MODIFY / VSET command.

When you do not explicitly specify a vset with a request, ON-Archive does not consider a vset with a status of N during automatic vset selection. When the vset status is U, ON-Archive asks the operator to check whether the required volumes are physically available. You can use the U (unknown) status when different volumes within a vset reside in different places.

PARAMETERS is reserved for future use.

PROTECTION specifies protection for the vset. You can use the PROTECTION qualifier to control the types of accesses that are performed on vsets. R indicates that ON-Archive permits read access, W indicates that ON-Archive permits write access, and D indicates that ON-Archive permits deletes. When you do not specify a PROTECTION qualifier, ON-Archive applies the default value of RWD to the vset. ON-Archive does not allow write access when the ON-Archive imports the vset. See "Access Permission" on page 2-27 for more information on access permissions.

IMPORTED indicates that ON-Archive imports a vset. An imported vset originates from a separate installation of ON-Archive. You must use the CATALOG command to identify an imported vset and its volumes to the ON-Archive catalog before you can access the data on them. See "What Is an Imported Volume Set?" on page 2-11. Also see "Initializing Volumes" on page 4-42 for an important note about imported volumes.

NOIMPORTED indicates that ON-Archive does not import a vset. It is the default.

When you define a vset, you *must* provide values for the following qualifiers:

- VSET
- CLASS
- DEVICE_TYPE
- DRIVER
- ACCESSIBILITY

The following list shows optional qualifiers. They offer additional features such as storage protection, temporary overflow storage area, description of location, interchange of data between two Universal Server database servers, and so on.

- PROTECTION
- PERMANENTLY_MOUNTED
- LOCATION
- TRANSIT
- IMPORTED

See "Working with Volume Sets and Volumes" on page 8-20 for information on commands that allow you to list, modify, and delete vsets.

DEFINE/VSET Examples

The following example defines a vset called **cust**, that is a system vset, consists of tape volumes, has high accessibility, and assigns the vset to a logical device called **tape**:

```
Onarchive>DEFINE/VSET=cust/CLASS=SYS/DRIVER=TAPE/ACCESS=0/DEVICE_TYPE=tape
```

The following example defines a vset called **test**, that is a user vset accessible only by users **pat** and **terry**. It is a disk vset assigned to a logical device called HO:

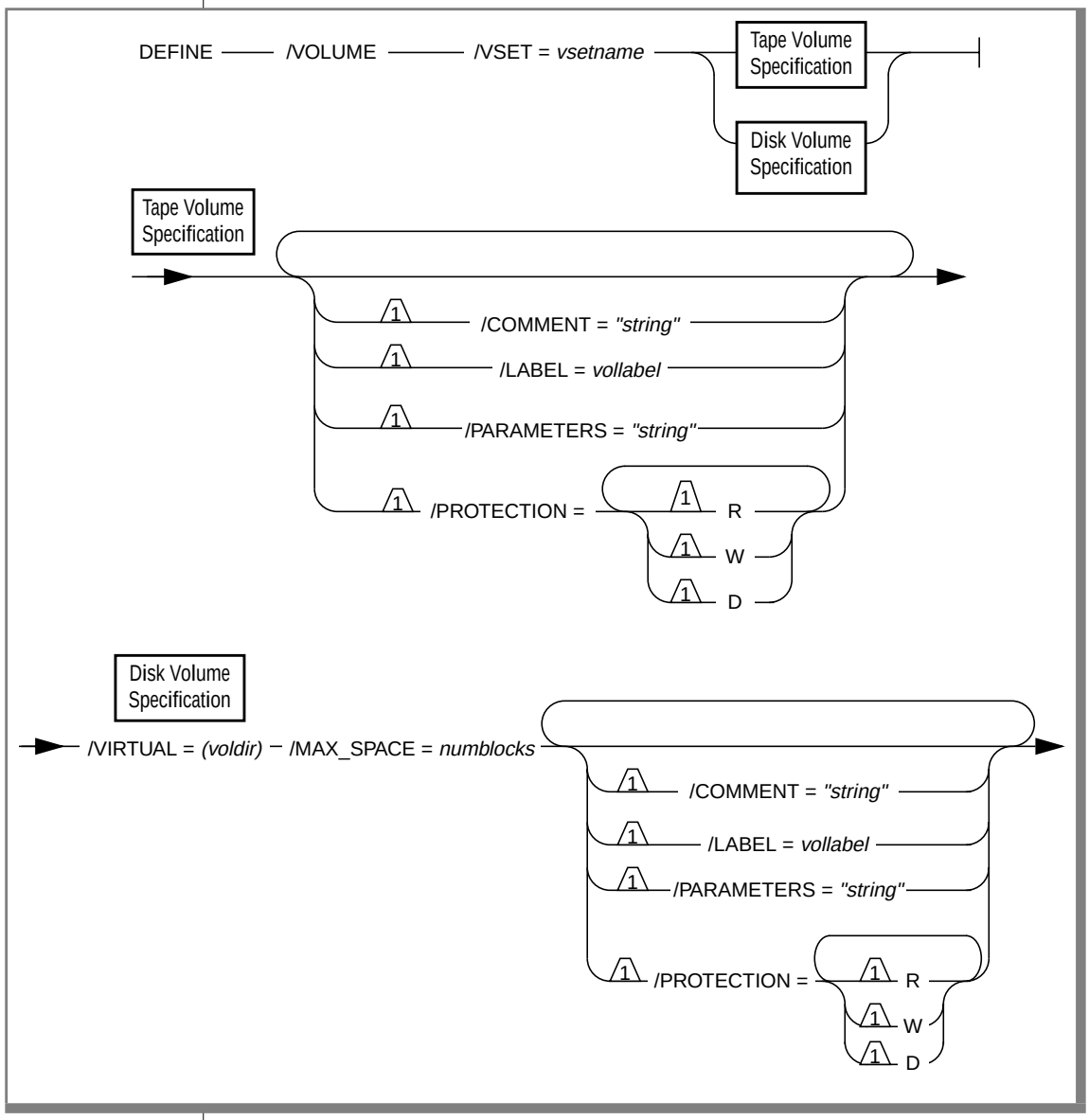
```
Onarchive> DEFINE/VSET=test/ACCESSIBILITY=30/DEVICE_TYPE=HO/CLASS=USER-
_Onarchive> /USER=(pat,terry)/DRIVER=DISK
```

The DEFINE/VOLUME Command

After you define a vset, you can use the DEFINE/VOLUME command to define one or more volumes for it. The defined volume inherits most of the attributes of the vset, such as location, device node, device type, density, and so on. See "What Is a Volume?" on page 2-13 for a description of a volume.

The following restrictions apply to the DEFINE/VOLUME command:

- When ON-Archive is running in OPERATOR privilege mode, only users **informix** and **root** can define a volume.
- When ON-Archive is running in GROUP privilege mode, only users in the **super_archive** group can define a volume. SYSTEM vsets have no other restrictions. The user name must reside in the access list for USER vsets.


DEFINE
/VOLUME
/VSET = vsetname
Tape Volume Specification
Disk Volume Specification
Tape Volume Specification
/COMMENT = "string"
/LABEL = vollabel
/PARAMETERS = "string"
/PROTECTION =
R
W
D
Disk Volume Specification
/VIRTUAL = (voldir)
/MAX_SPACE = numblocks
/COMMENT = "string"
/LABEL = vollabel
/PARAMETERS = "string"
/PROTECTION =
R
W
D

VOLUME — indicates that a volume definition comes next. ON-Archive assigns new volume numbers sequentially, starting with 1.

VSET — specifies the vset to which the volume belongs. The volume must be the same type of device (disk or tape) as the DRIVER qualifier defined for the vset.

vsetname is the name of a vset in the ON-Archive catalog.

COMMENT — a descriptive comment about the volume.

string is a text string. It cannot exceed 80 characters in length.

LABEL — specifies an ANSI-standard label for the volume.

vollabel is a string of up to six alphanumeric characters.

When you do not specify the LABEL qualifier, ON-Archive converts the volume number to ASCII and appends two blanks that automatically create a label for the volume. For example, the second volume in a vset could contain a label of '0002'. When you query the volume using the LIST/VOLUME command, and when ON-Archive requests that a you mount a volume, DEFINE/VOLUME displays the volume label.

PARAMETERS — is reserved for future use.

PROTECTION — specifies the volume protection. You can use the PROTECTION qualifier to control the accesses that are performed on volumes. R indicates ON-Archive permits read access, W indicates ON-Archive permits write access, D indicates ON-Archive permits deletes. When you do not specify a PROTECTION qualifier, ON-Archive applies the default value of RWD to the volume. See "Access Permission" on page 2-27 for more information on access permission.

Be aware that both volumes and vsets have access defined for them. The section "Volume Set and Volume Protection" on page 2-28 explains the rules for volume and vset access.

VIRTUAL — specifies the directory where a volume resides (the virtual definition of the volume).

voldir is a subdirectory of the directory defined as the physical device for the vset in the ON-Archive configuration file, **config.arc**. The directory specification must not contain any wildcard characters.

MAX_SPACE — This qualifier controls the maximum space (in blocks of 512 bytes) that ON-Archive allows the volume to use on a disk. ON-Archive stops writing on that volume when it reaches the maximum limit.

Note that the maximum space is not the same as the volume capacity. When the medium is already full, ON-Archive might run out of space before it reaches maximum space. You can use the MODIFY/VOLUME command to change the maximum space at any time.

numblocks is an integer between 0 and 99999999. The minimum usable size of MAX_SPACE is 16. This qualifier has no default value.

Unlike a vset, you do not assign a name to a volume. ON-Archive identifies a volume using its vset name and a volume number unique within the vset. ON-Archive automatically assigns the volume number when you define the volume.

You can use the MODIFY/VOLUME command to change a volume definition. See "The MODIFY/VOLUME Command" on page 8-42 for information on how to change a volume definition.

DEFINE/VOLUME Examples

The following example shows the simplest possible DEFINE/VOLUME command for a volume in a tape vset called **parts_vset**:

```
Onarchive> DEFINE/VOLUME/VSET=parts_vset
```

When ON-Archive successfully creates a volume, it displays the volume number it assigned as shown in the following sample message:

```
Volume 00004 defined for vset parts_vset.
Onarchive>
```

The following example defines a new volume for the disk vset **disk_arc**:

```
Onarchive>DEFINE/VOLUME/VSET=disk_arc/MAX_SPACE=10000-
_Onarchive> /VIRTUAL=(/home/disk_arc/vol1)
```

Initializing Volumes

ON-Archive initializes *non-imported* volumes that are part of a vset defined with DRIVER=TAPE or part of a vset defined with DRIVER=DISK and the NOPERMANENTLY_MOUNTED qualifier. ON-Archive does not initialize a volume that belongs to a vset defined with the IMPORTED qualifier.

When ON-Archive initializes a volume, it prompts you to select one of the physical devices defined for the logical device you specified for the vset. In the following example, ON-Archive prompts the operator to select one of two tape devices associated with the logical device:

```
Devices available for ONARCHIVE
DEVICE # DEVICE NAME
-------- -----------
 1       /dev/rst0
 2       /dev/rst1

Enter one of the device numbers (1-2) or (C)ancel : 1
```

To select the device, enter its number in the list. For example, you could enter 1 to select device **/dev/rst0**.

ON-Archive subsequently prompts you to mount a volume, as shown in the following line:

```
Please mount a volume on /dev/rst0 and press < RETURN > ** WRITE **
```

This message gives you time to mount a volume and also verifies your choice of a physical device.

When the volume has not already been formatted as an ON-Archive volume, ON-Archive prompts you with the following message before actually initializing the volume:

```
ARC-I-01098, Not an ONARCHIVE volume.

Initialize this volume?

(y/n) : y
Volume number 0001 created in parts_test
```

Enter y to initialize and create the volume.

When **onarchive** (not **onautovop**) needs to define a volume in the middle of executing a request, it prompts you with interaction similar to that shown in the preceding example. In general, however, you must define ahead of time any volumes you might need for an archive or backup.

Warning: *ON-Archive allows you to initialize volumes that already contain data, even though a restore could require that data. Do not initialize (define) a volume that could contain data needed for a restore because ON-Archive could initialize it and thus make it inaccessible.*

Disk Volumes

For vsets defined with DRIVER=DISK, you must define volumes using the VIRTUAL and MAX_SPACE qualifiers.

Creating a Subdirectory for a Disk Volume

Use the VIRTUAL qualifier to create a volume in a subdirectory of the directory you specified for the vset with the DEVICE_TYPE qualifier. You specify the directory pathname for the device using the DEVICE parameter in the ON-Archive configuration file, **config.arc**. See "DEVICE" on page 3-8 for information on the DEVICE configuration parameter. When the directory that the VIRTUAL qualifier specifies does not exist, ON-Archive automatically creates it.

For example, consider an ON-Archive configuration file, **config.arc**, with the following line:

```
DEVICE user = /usr/archive
```

You might define a disk vset called **user_vset** using this device with the following command:

```
Onarchive> DEFINE/VSET=user_vset/CLASS=SYSTEM/ACC=O/DRIVER=DISK/DEVICE=user
```

In the command to define volumes for this vset, the parameter for the VIRTUAL qualifier in the DEFINE/VOLUME command could be a subdirectory of **/usr/archive**, like **/usr/archive/vol**, as shown in the following command:

```
Onarchive> DEFINE/VOLUME/VSET=user_vset/VIRTUAL=(/usr/archive/vol)-
_Onarchive> /MAX_SPACE=10000
```

Size of Disk Volumes

Traditionally, as in the case of tapes, ON-Archive automatically limits the volume size to the capacity of the physical volume. In the case of disks, you can define a volume to occupy only a portion of the storage media. In fact, you must define a maximum space for each volume within a vset that you define with the DRIVER=DISK qualifier.

You can find the maximum space limit especially useful when other applications share the medium. You can use MAX_SPACE to prevent ON-Archive volumes from using too much space.

As explained in "Volumes in a Disk-Volume Set" on page 2-14, it is not advantageous to make numerous volumes on disk, so Informix recommends that you set MAX_SPACE to the amount of space you expect to devote to storing data in the vset. When you use an entire disk, you can set MAX_SPACE to 0, and ON-Archive uses all the space available. See the description of the MAX_SPACE parameter on page 4-41 for the minimum and maximum size of a disk volume.

When Can Volumes Be Reused?

ON-Archive allows you to reuse volumes only after you no longer need to restore all the data on the volumes. It prevents you from overwriting data that you might need in the future.

When you know that you do not need data on a volume for a restore, you can delete it, and thus make the volume available for reuse. You do not need to redefine the volume after you delete the data.

Deleting Data for Reuse

Once an archive or backup has outlived its usefulness you can delete it in the following ways:

- Automatic deletion

  Automatic deletion occurs when you specify an expiration date for the backup or archive request using the EXPIRY_DATE qualifier (before ON-Archive executes the request), and **onautovop** processes the expiration date. (See "Specifying When Archive Data Is Obsolete" on page 5-18.)

- Manual deletion

  You can use the ON-Archive REMOVE/REQUEST command to remove stored data from the catalog. (See "Removing Save Sets to Reclaim Storage Space" on page 8-19.)

A deletion operation requires two-steps. The first step logically deletes the stored-data information in the catalog. This deletion occurs when ON-Archive processes the expiration date, or when ON-Archive issues the REMOVE/REQUEST command.

The second step removes the data from the volume. This step is different for save sets on disk and tape volumes. The section "Removing Save Sets to Reclaim Storage Space" on page 8-19 explains this step.

How ON-Archive Selects Volume Sets, Devices, and Volumes

When ON-Archive must choose the most available vset, it uses the following criteria in the following order:

1. It uses only on-site and *location-unknown* vsets. (See the description of the LOCATION qualifier, on page 4-36, for the DEFINE / VSET command.)
2. It uses USER class vsets before SYSTEM vsets, when the user executing the request resides on the access list. (See the descriptions of the CLASS qualifier, on page 4-34, and the USER qualifier, on page 4-34, for the DEFINE / VSET command).
3. Vsets with low-accessibility-level numbers are considered more available than vsets with higher-accessibility-level numbers. (See the description of the ACCESSIBILITY qualifier, on page 4-33, for the DEFINE / VSET command.)

See "Volume Sets and Volumes for onautovop" on page 4-53 for more information on how **onautovop** selects vsets and volumes.

Which Physical Device Is Used?

When the vset for a request specifies a logical device in the **config.arc** file that has more than one physical device, **onarchive** and **onautovop** use different methods to determine the correct device to use.

The **onarchive** utility prompts users to select one of the possible devices for the request.

The virtual operator (**onautovop**), on the other hand, checks the devices in the order listed in the **config.arc** file until it finds an available volume in the vset for the request.

Using Volume Sets That Have Remote Devices

When you use a vset that has the DEVICE_TYPE defined as a logical device that uses a remote tape device, you must specify the APART qualifier in your archive command as shown in the following line:

```
Onarchive> ARCHIVE/DBSPACESET=frequent/VSET=rem_rot1/APART
```

ON-Archive creates all save sets using remote tape devices on separate volumes.

See "Using Remote Devices" on page 3-13 for information on how to define a remote device.

Which Volumes Are Used?

As described in "Can Data Be Placed on a Specific Volume?" on page 2-14, you do not specify a volume when you create a request. Instead, **onarchive** uses the next volume in the vset that is not already full and (in the case of tape volumes) not already in use. You can use disk volumes at the same time because ON-Archive writes save sets as files within the volume directory. Thus, the next volume in a disk vset is simply the next volume (that can include the current volume) that is not already full.

When a volume fills during the execution of a request, **onarchive** prompts the operator to mount a new volume. When no other defined volume is available, it prompts the operator through the creation of a new volume.

As described in "Which Volumes Does onautovop Use?" on page 4-54, when a volume fills when **onautovop** executes a request, **onautovop** continues the save set on another available volume in the vset. When no other volume is defined and available (and the request does not specify that you use a transit vset), the request status changes to FAILED.

Emergency Situations

Occasionally, you might need to use ON-Archive when you cannot use the catalog. This can occur in the following situations that cause Universal Server to stop processing:

- When the disk containing the root dbspace (and thus the **sysmaster** database and the ON-Archive catalog) fails.
- When the disk containing your physical-log file or logical-log files fails.
- When your logical log fills.

You cannot use **onarchive** in these situations because **onarchive** accesses the ON-Archive catalog through Universal Server.

When you cannot use the ON-Archive catalog, you can use **ondatartr** to perform physical and logical restores and back up your logical-log files.

The **ondatartr** utility is described in "The ondatartr Utility" on page 9-12. See "Performing Emergency Logical-Log File Backups" on page 6-13, "Perform a Cold Physical Restore" on page 7-27, and "Perform a Logical Restore with ondatartr" on page 7-30 for instructions on using **ondatartr** in emergency situations.

Defining an Emergency Volume Set and Volume

The **ondatartr** utility *creates* an emergency vset and volume in the following situations:

- When it performs emergency log backups
- When it salvages logical-log files during a cold physical restore

See "Performing Emergency Logical-Log File Backups" on page 6-13 for information on emergency logical-log file backups. Also see the description of the SALVAGELOGS qualifier for the **ondatartr** "RETRIEVE / DBSPACESET Command" on page 9-24 and "Steps to Restore the Whole System" on page 7-22 for information on when **ondatartr** salvages logical-log files.

You can *define* an emergency vset and volume ahead of time as placeholders in the ON-Archive catalog. After **ondatartr** performs an emergency logical-log backup or salvages logical-log files during a restore, you must *catalog* the save sets before you can use them. The **ondatartr** utility *creates* the vset and volume that it needs when it performs the backup or salvage operation. Defining the vset and volume ahead of time reduces the number of things you must do before you can catalog these volumes.

The **ondatartr** utility always names the vset it creates ONDATARTRLOG and always creates volumes in the vset starting with volume number 1. Therefore, use the following guidelines to define an emergency vset and volume:

- Name the vset ONDATARTRLOG.
- Specify the same type of device (disk or tape) that **ondatartr** uses to back up or salvage the log files.
- Define the number of volumes that **ondatartr** needs to use (generally, only one). (The vset only stores logical-log files in an emergency, so you never store more than the amount of space devoted to the entire logical log.)

Emergency Disk Volumes

For an emergency backup of logical-log files, you might find it convenient to use disk volumes for emergency data. The risk of losing data on disk (as opposed to tape) is small because you almost immediately copy data stored in the emergency vset and volume to another vset.

When you decide to use disk volumes, create a new, appropriately named directory for the volume. For example, when the logical device called **arc_disk** maps to a directory called **/archive**, you might create a subdirectory called **em_logs** and define the vset and volume in the following manner:

```
% mkdir /archive/em_logs
% onarchive
Onarchive> DEFINE/VSET=ONDATARTRLOG/ACCESS=O/DEVICE_TYPE=arc_disk/CLASS=system-
_Onarchive> /DRIVER=disk/PROTECT='RD'
Onarchive> DEFINE/VOLUME/VSET=ONDATARTRLOG/VIRTUAL=(/archive/em_logs)/MAX=4000
```

The largest MAX_SPACE needed for the emergency volume equals the size of the entire logical log. Remember that MAX_SPACE is expressed in blocks, each 512 bytes in length. The space remains unused until ON-Archive writes data to the volume.

Emergency Tape Volumes

When you use tape volumes, you must mount an unused tape for each volume you define. You can later redefine the tape for another vset because you do not use the tape for the logical-log backup or salvage operation. You only need it to define the volume in the catalog. When **ondatartr** performs the emergency backup or salvage operation, you can supply any blank tape to receive the data.

Using Unattended Operations

Perform the following steps to request an unattended operation:

- Create a request that the virtual operator can execute using **onautovop**.
- Start **onautovop** at a specified time.
- Ensure that the vsets and volumes needed for the unattended operation remain available for the virtual operator.
- Monitor success or failure of the operations.

When **root** or **informix** starts **onautovop** (and ON-Archive is in OPERATOR privilege mode) or a user in the **super_archive** group starts **onautovop** (while running in GROUP privilege mode), **onautovop** executes all pending requests. When any other user starts **onautovop**, **onautovop** only executes the pending requests that user submitted.

The **onautovop** program takes the following actions once it starts:

- It attempts to execute all eligible and pending requests. It executes them one at a time, starting with the lowest request id. An eligible and pending request meets the following criteria:
  - All the wait conditions have been met.
  - Its status is NEW. (The status must have been NEW before **onautovop** began executing.)
- It identifies save sets with elapsed expiration dates, removes them from the catalog, and marks them for physical deletion.

After it completes these operations, **onautovop** terminates.

When you start more than one **onautovop** process at the same time, the first **onautovop** to start executes the oldest eligible request in the catalog, changing its status to EXECUTING. The second **onautovop** process executes the next eligible request, that originally was the second-oldest request.

Creating Requests for onautovop

You can use the following four command qualifiers when you create requests for execution using **onautovop**:

- VSET
- REPEAT
- TRANSIT
- NOTIFY

The following sections describe each of these qualifiers.

Using an Appropriate Volume Set

The **onautovop** process only uses devices that identified with a logical device name ending with the string _VOP, as shown in the following example:

```
DEVICE TAPE_VOP = /dev/rst0, /dev/rst1
DEVICE QIC_VOP = /dev/rst8
```

Tip: *You can prevent **onautovop** from executing certain requests. To do this, specify that the requests reside in a vset that does not use a _VOP device.*

Using REPEAT to Generate New Requests

The REPEAT qualifier enables you to automatically generate a new request as a by-product of the request you are executing. The REPEAT qualifier automatically creates a new request that does essentially the same thing as the original request. (See "Wait and Repeat Qualifiers" on page 8-82.)

For example, the following BACKUP request creates a new BACKUP request after the backup completes. In this case, the REPEAT qualifier specifies that the new request begin executing 20 minutes after the original request.

```
Onarchive> BACKUP/LOGFILE/AUTOMATIC/VSET=logvset/REPEAT=0-0:20:00
```

To stop a repeating request, use the MODIFY/COMMAND command to remove the REPEAT qualifier before the request executes again.

Using TRANSIT to Assign Overflow Space

When the volume or volumes in a vset could become full during unattended operations, you might want to specify a *transit vset*, as shown in the following example, to assign overflow space:

```
Onarchive> BACKUP/LOGFILE/AUTOMATIC/VSET=logvset/TRANSIT/REPEAT=0-0:20:00
```

The TRANSIT qualifier does not mean that the save set of log files is immediately placed in a transit volume; ON-Archive writes the log files only to the transit vset when the volumes in the regular vset (**logvset**, in this example) become full or unavailable.

See "What Is a Transit Volume Set?" on page 2-11 for a description of a transit vset. See "TRANSIT" on page 4-35 for information on how to create a transit vset. See the TRANSIT qualifier under "Archive and Backup Qualifiers" on page 8-70 for information on selecting a transit vset.

Using NOTIFY to Inform the Operator

See "Monitoring Unattended Operations" on page 4-54 for a description of how to use the NOTIFY qualifier.

Starting onautovop

Use the **cron_autovop** utility to start **onautovop** automatically at specified times. See "The cron_autovop Utility" on page 9-5 for instructions on using **cron_autovop**.

See "The start_autovop Script" on page 9-31 for information on starting **onautovop** manually. Although **onautovop** stops once it has carried out all its tasks, you can also stop it manually. See "The stop_autovop Script" on page 9-33 for information on stopping **onautovop** manually.

Volume Sets and Volumes for onautovop

When **onarchive** executes a request, it relies on the operator to do the following things:

- Choose the physical device for a request
- Mount (and define) new volumes, when necessary

When **onautovop** executes a request, it does not rely on a human operator to make decisions about the physical device to use, or to mount volumes at appropriate times. It cannot define new volumes when they are needed.

Which Device Does onautovop Use?

When a logical device specifies more than one physical device, **onautovop** does not prompt an operator for the device to use. Instead, it searches through all the devices in the order listed in the ON-Archive configuration file, **config.arc**. When **onautovop** finds a volume that it can use for the request (in the right vset, and available for use), it uses the device for that volume.

When the device list specifies a remote device, Informix recommends that you verify beforehand that **onautovop** can access the device and read the tape. To do this, enter a command on the UNIX command line to read the tape. For example, the following command attempts to read the tape device **/dev/rmt/3m** on host **river**:

```
% rsh river dd if=/dev/rmt/3m count=1
```

The output goes to the standard out file. When you receive an error from the operating system, you need to fix the error before **onautovop** can read the device.

Which Volume Set Does onautovop Use?

As mentioned in "Using an Appropriate Volume Set" on page 4-51, you must use names that end in _VOP to define vsets used by **onautovop** in **config.arc**.

Beyond the requirement to terminate the vset name with VOP, **onautovop** adheres to the same procedure as **onarchive** for selecting a vset. The section "How ON-Archive Selects Volume Sets, Devices, and Volumes" on page 4-46 describes this procedure.

Which Volumes Does onautovop Use?

Unlike **onarchive**, when **onautovop** does not have a volume available for the appropriate vset, it does not prompt the operator to mount a new volume. The **onautovop** process picks the first volume that is not full. When it runs out of space while writing to the volume, it picks the next volume in the vset that is not full. When you run multiple **onautovop** processes that use the same vset, each process in turn, picks the next volume in the vset that is not full. See "The LIST/VOLUME Command" on page 8-36 for information on how to list each of the volumes in a vset and the amount of space that has been used on each one.

When you cannot find any available volumes on any of the devices specified by the request when **onautovop** executes it (and the request does not use a transit vset), the status of the request is reset to NEW, and the request does not execute.

When the save set spans two or more volumes, and one of the required volumes is not available (and the request does not use a transit vset or the transit vset is filled up), the request status is set to FAILED.

Monitoring Unattended Operations

You could also include the NOTIFY qualifier in a request executed by **onautovop.** The NOTIFY qualifier, illustrated in the following example, reports the outcome of a request:

```
Onarchive> ARCHIVE/DBSPACESET=*/VSET=archive/TRANSIT/REPEAT=7/NOTIFY
```

When **onautovop** executes a request that includes the NOTIFY qualifier, **onautovop** sends electronic mail to the user who created the request to report the outcome. Because NOTIFY sends electronic mail to the user who created the request, Informix recommends that the same user creates all the requests that **onautovop** executes.

Chapter

5

Creating an Archive

This chapter explains how to plan for and create archives of your Universal Server data. It also describes in detail how ON-Archive works with Universal Server to create an archive.

This chapter discusses the following topics:

- Considerations before you create an archive
- Defining dbspace sets
- Creating archives using the ARCHIVE command
- Scheduling archives
- Details of what happens during an archive

Before You Create an Archive

You must consider several things when you plan how to archive your Universal Server data. This section addresses the following topics:

- What Universal Server data is copied during an archive
- The three levels of archives
- Tasks you must perform before you create an archive

What Data Is Archived?

All the data that Universal Server manages is archived with the following exceptions:

- Dbspace pages allocated to Universal Server but not yet allocated to a tblspace extent are not archived.
- None of the configuration files are archived.
- Pages from mirror chunks are not archived when the corresponding primary chunks are accessible.
- Blobs in blobspaces stored on optical platters (managed with INFORMIX-OnLine / Optical) are not archived.
- Pages in temporary dbspaces are not archived.

Tip: *Make sure you archive your critical dbspaces first to make the archiving process easier.*

Full-System Archives and Dbspace-Set Archives

ON-Archive gives you the option of archiving a subset of your Universal Server data. To do this you create sets of dbspaces to archive together. See "What Is a Dbspace Set?" on page 2-19.

A *full-system* archive archives all of the data managed with your Universal Server database server, with the exceptions previously listed. A full-system archive could take some time to create, depending on the how much data you must back up. A full-system archive provides a point of reference for subsequent archives and for logical-log backups.

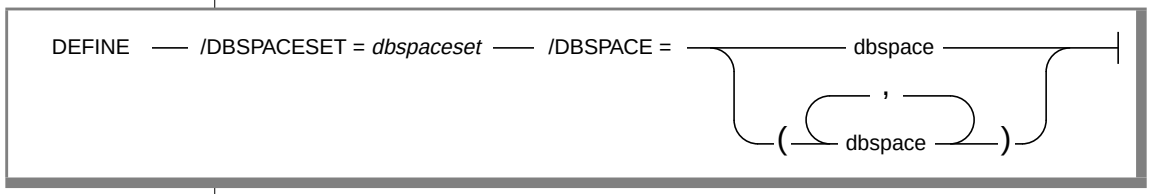
You use the DEFINE / DBSPACESET command, described in the following section, to define a dbspace set.

The DEFINE/DBSPACESET Command

The DEFINE / DBSPACESET command groups specified dbspaces into a *dbspace set*. Creating a dbspace set enables you to conveniently archive multiple dbspaces simply by specifying the name of the set rather than specifying each dbspace individually.

The following restrictions apply to the DEFINE / DBSPACESET command:

- When ON-Archive is running in OPERATOR privilege mode, only users **root** and **informix** can define dbspace sets.
- When ON-Archive is running in GROUP privilege mode, only users in the **super_archive** group can define dbspace sets.



DBSPACESET specifies the name of the new dbspace set. When you want to archive all the dbspaces managed with Universal Server in one set, you do not define a dbspace set; instead, use the * parameter with the DBSPACESET qualifier in the ARCHIVE command.

dbspaceset is the name of the new dbspace set. You must use a unique name that cannot exceed 18 characters in length.

DBSPACE specifies the names of the dbspaces and / or blobspaces to include in the dbspace set.

dbspace must exist as a dbspace or blobspace. A dbspace or blobspace can belong to more than one dbspace set.

DEFINE/DBSPACESET Example

The following command defines a dbspace set called **frequent**, composed of the **root** dbspace and a dbspace called **freq_use**:

```
Onarchive>
DEFINE/DBSPACESET=frequent/DBSPACE=(root,freq_use)
```

See "The ARCHIVE Command" on page 5-12, "The RETRIEVE / DBSPACESET Command" on page 7-14, "The LIST / DATABASE Command" on page 8-49, "The LIST / DBSPACESET Command" on page 8-52, and "The DELETE / DBSPACESET Command" on page 8-48 for more information on using dbspace sets.

What Are Archive Levels?

When you store a lot of data that Universal Server manages, it does not always make sense to archive all the data all the time. For example, when some of your information changes quite a bit, but some remains very stable, it seems inefficient to archive the stable information every time you archive the volatile information.

To provide a more flexible archive environment, Universal Server supports three types of incremental archives, creating the following archive *levels*:

Level-0 archives all used pages.

Level-1 archives all changes after the last level-0 archive.

Level-2 archives all changes after the last level-1 archive.

These three archive levels are explained in the following sections.

See "Scheduling Archives" on page 5-20 for advice on scheduling archives.

Level-0 Archives

A level-0 archive is the baseline archive. It contains a copy of every used disk page (dbspace and blobspace) that needed to restore the Universal Server database server to its state at that time. When an event (such as a fire or flood) completely destroys a computer, you must use a level-0 archive to completely restore Universal Server data on the replacement computer.

A level-0 archive can consume lots of time because you must write all the pages to tape. Level-1 and level-2 archives take less time to create because only the changes to the data are written to tape. However, when you create level-0 archives infrequently, the level-1 archive could become quite large. For example, when the last level-0 archive was completed a day ago, and few changes occurred after the archive, the level-1 archive could remain small. However, when the last level-0 archive occurred a month ago, and many changes occurred after the archive, the level-1 archive could grow very large.

For on-line archives, the data on the archive tape reflects the contents of the dbspaces and blobspaces at the time the level-0 archive began. (The time the archive started could reflect the last checkpoint before the archive started, providing that no database activity took place between the checkpoint and the archive.)

Level-1 Archives

A level-1 archive contains a copy of every changed page containing data and system-overhead information after the last level-0 archive. All data copied to the archive reflects the state of the data at the time the level-1 archive began. A level-1 archive usually takes less time than a level-0 archive because only part of the data that Universal Server manages gets copied to the archive tape.

Level-2 Archives

A level-2 archive contains a copy of every changed page containing data and system overhead information after the last level-1 archive. All data copied to the archive reflects the state of the data at the time the level-2 archive began.

A level-2 archive after a level-1 archive usually takes less time than another level-1 archive because only the changes made after the last level-1 archive (instead of the last level-0 archive) are archived.

Preliminary Tasks

Before you can create an archive using ON-Archive, at the very least you must complete the following tasks:

- Start **oncatalgr** and **onarchive**. (See "Starting the Cataloger" on page 4-5 and "Starting onarchive" on page 4-7.)
- Define dbspace sets when you are archiving dbspaces selectively. (See "The DEFINE / DBSPACESET Command" on page 5-5.)
- Create a volume set with a sufficient number of volumes. (See"Using Volume Sets and Volumes" on page 4-31.)
- Understand how to create and execute ON-Archive requests. (See "Creating and Executing Requests" on page 4-26)

In addition, however, you must also consider the following tasks before you create an archive:

- Make sure you allot sufficient logical-log space to create an archive.
- Keep a copy of your ONCONFIG file.
- Verify data consistency.
- Ensure that Universal Server operates in the appropriate mode.
- Plan for operator availability.
- Synchronize with other administrative tasks.
- Do not use background mode.
- Label tapes appropriately.

The following sections address each of these topics.

Make Sure You Have Enough Logical-Log Space

When the total available space in the logical log (all the logical-log files) is less than half of a single log file, Universal Server does not create an archive. You must back up the logical-log files and attempt the archive again.

You cannot add a logical-log file or mirroring during an archive.

When only one tape device is available, make sure all your logical-log files are backed up before you start your archive to reduce the likelihood of filling the logical log during the archive.

Copy Your ONCONFIG File

Keep a copy of the current ONCONFIG file when you create a level-0 archive. You need this information to restore Universal Server data from the archive tape.

Verify Consistency Before a Level-0 Archive

To ensure the integrity of your archives, periodically verify that all Universal Server data and overhead information is consistent before you create a level-0 (full-system) archive. You need not check for consistency before every level-0 archive. Informix recommends, however, that you save such an archive until the next time you create an archive after verifying Universal Server consistency. See the *INFORMIX-Universal Server Administrator's Guide* for information on consistency checking.

Choose a Universal Server Mode

You must create archives while Universal Server operates in on-line mode or quiescent mode. When you use **onarchive** to start the archive, the terminal that you use to initiate the command is dedicated to the archive (for displaying messages) until it is complete. Once you start an archive, you must not change the Universal Server mode until the archive is finished; changing the mode terminates the archive activity.

On-line Mode

An on-line archive is an archive that is created while Universal Server is in on-line mode. This type of archive is convenient when you want your Universal Server database server to remain accessible while you are creating the archive.

Some minor inconveniences are associated with on-line archives. An on-line archive can slow checkpoint activity, and that can contribute to a loss in performance. However, this decline in performance is far less costly than the time that you could lose when you deny Universal Server access to users during an archive.

During an on-line archive, allocation of some disk pages in dbspaces and blobspaces could temporarily freeze. Disk-page allocation in dbspaces and blobspaces is blocked for one chunk at a time until the used pages in the chunk are archived.

Quiescent Mode

A quiescent archive is created while Universal Server operates in quiescent mode. Quiescent archives are useful when you want to eliminate partial transactions in an archive.

Quiescent archives could be impractical when users need continuous access to the databases that Universal Server manages.

Ensure That the Operator Is Available

When you are making a multivolume on-line archive using **onarchive**, make sure an operator is on duty to mount tapes as prompted.

An archive could take several reels of tape. When an operator is not available to mount a new tape when one becomes full, the archive waits. During this wait, when the archive is an on-line archive, the physical log space could fill up, and make Universal Server abort the archive. Thus, Informix recommends that you keep an operator available.

When you are using ON-Archive, use unattended operations to avoid this situation, as described in "Using Unattended Operations" on page 4-50, and mount appropriate volumes on multiple devices prior to the time that you need them.

Synchronize Administrative Tasks with Archives

The following administrative changes require a level-0 archive as part of the procedure. Consider waiting to make these changes until your next regularly scheduled level-0 archive.

- Changing TAPEDEV or LTAPEDEV from **/dev/null** requires an archive after the change is made.
- Adding logging to a database requires an archive after logging is added.

- Adding a dbspace or blobspace requires that you archive the space before restoring it with anything less than a full-system restore.
- Starting mirroring for a dbspace that contains logical-log files requires an archive after the change to initiate mirroring.
- Adding a logical-log file requires an archive afterward to make the log file available.
- Dropping a logical-log file requires an archive after the log file is dropped.
- Moving one or more logical-log files requires an archive after the logical-log file is dropped and after it is added.
- Changing the size or location of the physical log requires an archive after shared memory is reinitialized.
- Dropping a chunk requires an archive before you can reuse the dbspace containing that chunk.

You can archive only the root dbspace (instead of all dbspaces) for the following administrative changes:

- Adding mirroring
- Adding a logical-log file
- Changing the size or location of the physical log
- Dropping a chunk

When you add logging for a database, the logging change takes effect after all dbspaces containing data in the database become archived. The dbspaces do not require archiving all at once; Universal Server checks after each archive to see when you can initiate logging.

Label Tapes

When you need to perform a data restore, it is very useful (when not critical) to know the data contents of each tape. Thus, Informix recommends that you place labels on your archive and backup tapes when they are created. When your tapes are not large enough to hold all the information needed, you could create a tape log book to track tape information.

When you label tapes that ON-Archive creates, you must record the following information on the label:

- Volume set (vset) name
- Volume number
- Level of archive
- Date and time
- Dbspaces archived
- Save set id (request id)
- Sequence of tape in the save set (which you might not know until the archive or backup operation is finished)

The following example shows what a label can look like:

```
Disk1_vset, Vol 8
Level 0:Dbspaceset_1:Saveset 45:Wed Nov 27, 1993 20:45
Tape 1 of 3
```

Creating an Archive

After you complete the preliminary tasks described in the previous section, you are ready to create an archive.

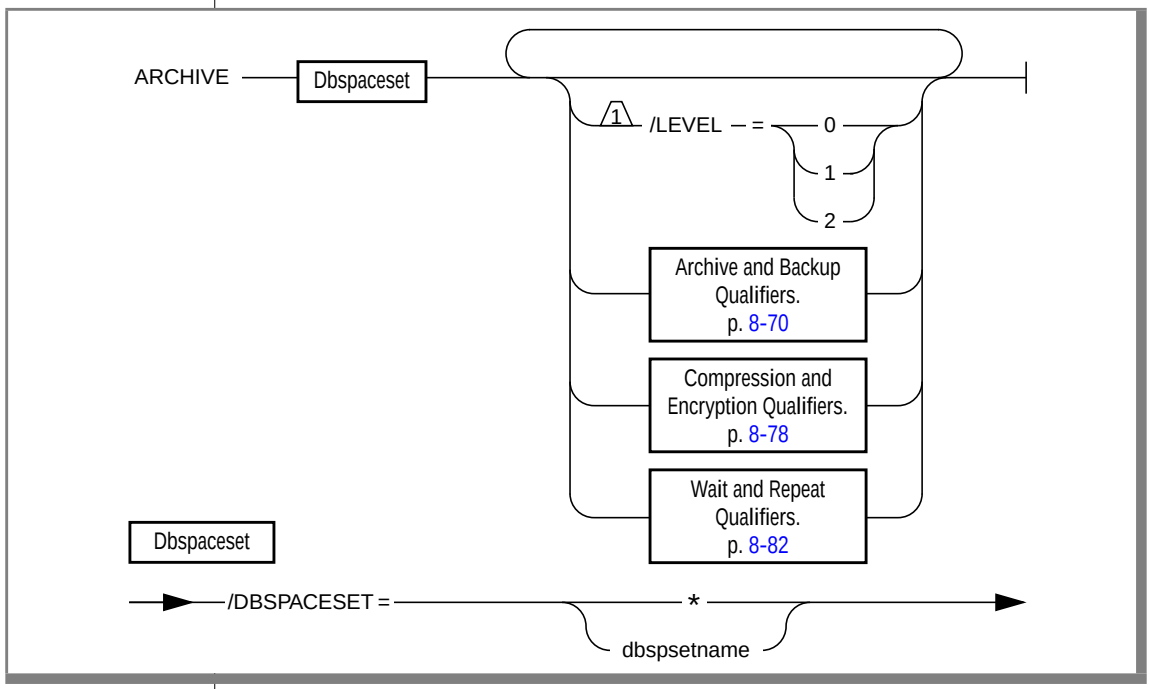
The ARCHIVE Command

The ARCHIVE command creates a request that, when executed, creates an archive. It can create an archive of all the data managed with your INFORMIX-Universal Server (a full-system archive), or an archive of specified dbspace sets. See "What Data Is Archived?" on page 5-4 for a description of the data that is backed up during an archive.

The following restrictions apply to the ARCHIVE command:

- When ON-Archive is running in OPERATOR mode, only users **root** and **informix** can create requests to archive data.
- When ON-Archive is running in GROUP mode, only users in the **super_archive** group can create requests to archive data.

Once the requests are created, restrictions also apply to their execution. See "The EXECUTE Command" on page 4-27 and "The EXECUTE / VSET Command" on page 8-33 for more information on these restrictions.



LEVEL specifies what level archive to create. Archive levels are explained in "What Are Archive Levels?" on page 5-6. When you do not specify LEVEL, ON-Archive assumes a level-0 archive.

- 0 specifies a level-0 archive.
- 1 specifies a level-1 archive.
- 2 specifies a level-2 archive.

DBSPACESET specifies the dbspace set or sets that you want to archive. When * is specified, you archive (a full-system archive) all the dbspaces and blobspaces managed with Universal Server.

dbspsetname must already be defined, using "The DEFINE / DBSPACESET Command" on page 5-5.

ARCHIVE Examples

The following example creates a request to perform a level-0 archive of all the dbspace sets managed with ON-Archive:

```
Onarchive> ARCHIVE/DBSPACESET=*
```

The following example creates a request to perform a level-1 archive of the dbspace set called **crit_set**, on a blank tape.

```
Onarchive> ARCHIVE/DBSPACESET=crit_set/APART
```

Remote Archives

A remote archive gets written to a tape device attached to another computer. You cannot initiate an archive from a remote computer; you can only use a tape device on a remote computer as a destination for an archive.

Using a Remote Tape Device

Regardless of the tool you use, when you use a remote tape device, you must ensure that the user creating the archives can run an operating system shell on the remote computer from their computer. The shell run is the same as the shell that started the remote operation, unless you specify a different shell with the **DBREMOTECMD** environment variable.

As your operating-system documentation must explain, you can set up access control in your networked environment to give users access to remote devices in the following three ways:

- To allow everyone on one computer (for example, a computer called **local**) access a different computer (called, for example, **remote**), you could put the name of the first computer into the **/etc/hosts.equiv** file on the second computer. In this example, **local** resides in the **hosts.equiv** file on **remote**.
- To give only specific users access to the second computer (for example, allowing user **informix** on **local** to access **remote**), you put the name of the second computer in the **~/.rhosts** file for that user. The user must own the **.rhosts** file.
- To let **root** access the other computer, the name of the first computer (**local**) must reside as an entry in the file called **/.rhosts** on the other computer (**remote**).

See "DEVICE" on page 3-8 for more information about using remote devices.

Using ARCHIVE Options

When you create your archive requests, you could take advantage of the following archiving capabilities:

- Making an archive of all the data
- Making archives of different levels
- Specifying the vset to use
- Keeping archives on separate tapes
- Creating multiple copies of archives
- Setting expiration dates for the data
- Using data compression or encryption
- Sending notification of archive-request completion

The following sections explain each of these capabilities.

Making a Full-System Archive

To make a full-system archive of the data managed with your Universal Server database server, specify an * as the parameter for the DBSPACESET qualifier in the ARCHIVE command. The following example requests a full-system archive:

```
Onarchive> ARCHIVE/DBSPACESET=*
```

You cannot define a dbspace set with the * parameter, so ON-Archive creates a special dbspace set when you use this command.

You must archive in this way when you want to perform a full-system restore (using the * parameter).

Making Archives of Different Levels

When you do not specify an archive level in your archive request, ON-Archive creates a level-0 archive. To create a level-1 or level-2 archive, you must specify the level in your request, as shown in the following example:

```
Onarchive> ARCHIVE/DBSPACESET=*/LEVEL=1
```

Specifying Volume Sets for Archives

When you define more than one vset, as described in "Using Volume Sets and Volumes" on page 4-31, presumably you did this to ease the task of managing vsets and volumes. In that case, make sure to direct your archive data to the appropriate vset using the VSET qualifier, as shown in the following example.

```
Onarchive> ARCHIVE/DBSPACESET=frequent/VSET=rot1vset
```

When you do not specify a vset, ON-Archive automatically chooses the most available vset, as described "How ON-Archive Selects Volume Sets, Devices, and Volumes" on page 4-46.

Keeping Archives on Separate Volumes

To simplify tape handling, you could put each archive (save set) on a separate volume. You can use the APART qualifier to do this, as shown in the following example:

```
Onarchive> ARCHIVE/DBSPACESET=frequent/VSET=rem_rot1/APART
```

The request does not execute until a blank volume becomes available. Once you write the save set (it still could span more than one volume, all of which can only contain that save set), you cannot write any other save sets to its volume (or volumes).

Creating Multiple Copies of an Archive

ON-Archive normally makes one copy of each save set. You could create multiple copies of archives; you can use them in the event of media corruption. When one volume or vset gets corrupted, you can retrieve the data it contains from the second copy stored on another vset.

You can create multiple copies of archives in the following ways:

- Creating copy requests using the COPIES qualifier
- Creating copies with the COPY command

The following sections explain each of these methods.

Creating Copy Requests Using the COPIES Qualifier

To make ON-Archive create requests for copies of archives, specify the COPIES qualifier. Different copies of the same save set are never kept on the same vset. So, for each copy you specify, you must include the name of a different vset.

The following example creates a request for an archive and requests a copy using the COPIES qualifier:

```
Onarchive>
ARCHIVE/DBSPACESET=frequent/COPIES=2/VSET=(rot1,rot1_bkup)
```

ON-Archive chooses the most available vset from the lists of vsets and makes the original archive on that vset. (See "How ON-Archive Selects Volume Sets, Devices, and Volumes" on page 4-46.) It then generates a child copy request for each of the other copies, directing them to the other specified vsets.

You must explicitly execute the child requests, using one of the methods described in "Creating and Executing Requests" on page 4-26. The status of the parent request does not change to COMPLETED until all of the child requests are also completed.

Creating Copies with the COPY Command

You can use the COPY/VSET command (described in "The COPY/VSET Command" on page 8-24) to create copies of archives once they exist. You can copy an individual save set, provided you possess read access to the vset and volume. When you are **informix** or **root**, you can copy a whole vset.

The following command creates a request that (when executed) copies the archive save set on a vset called **rot1** created with request id 68 to a vset called **rot1_bkup**:

```
Onarchive> COPY/VSET=rot1/DESTINATION=rot1_bkup/REQUEST=68
```

Specifying When Archive Data Is Obsolete

To reclaim storage space automatically, define an expiration date in your archive request, as described in "When Can Volumes Be Reused?" on page 4-45. The following example makes the volume containing this save set reusable after 21 days:

```
Onarchive>
ARCHIVE/DBSPACESET=frequent/VSET=rot1/APART/EXPIRY_DATE=21
```

Only **onautovop** processes the expiration date, so you must make sure that **onautovop** runs after the expiration date to use the volume.

You do not need to specify the APART qualifier when you use EXPIRY_DATE but, as described in "When Can Volumes Be Reused?" on page 4-45, when any save set on a tape volume is not marked as obsolete, you cannot reuse the volume. Specifying APART ensures that the volume contains only the one save set and is reusable after the expiration date.

You also can remove completed requests from the catalog to remove obsolete save sets, as explained in "Removing Save Sets to Reclaim Storage Space" on page 8-19.

Using Data Compression or Encryption

You could compress and/or encrypt your archive data as it is written to a volume. Before using compression or encryption, make sure to read about them in the following sections:

- "Data Compression" on page 2-24
- "Data Encryption" on page 2-30

To see the data compression and encryption methods available, use the LIST/METHOD command described on page 8-66.

Obtaining Notification and Logging Results

The LOG and NOTIFY qualifiers enable you to receive feedback from ON-Archive about the result of executing a request.

When you want to keep a running record of the output of requests, you can use the LOG qualifier to send output to a file, as shown in the following line:

```
Onarchive> ARCHIVE/DBSPACESET=*/VSET=system/LOG
```

When you execute many archive requests, or create automatically executed requests, Informix recommends that you receive notification through the mail after an archive request executes. You can use the NOTIFY qualifier to instruct ON-Archive to send mail to the person who created the request (not the user who executed the request), as shown in the following line:

```
Onarchive> ARCHIVE/DBSPACESET=frequent/VSET=rot1/NOTIFY
```

You can use NOTIFY and LOG in combination.

Universal Server Message-Log File

You could see messages in the Universal Server message-log file that indicate when **onarchive** starts an archive and when the archive ends. You could see both messages in the Universal Server message log file, however, and receive a FAILURE status for the archive.

When you see both messages, it means that the archive data has been written to the tape or the disk. ON-Archive then goes on, however, to validate the tape (by rereading it) and update the ON-Archive catalog with the file information. When either of these activities results in failure, **onarchive** marks the status of the archive as failed.

When **onarchive** failed in writing the file information to the ON-Archive catalog, you can run **arc_purge.sql** to purge any incomplete information from the catalog and then catalog the archive volumes using the CATALOG command.

Scheduling Archives

You must make a regular schedule for creating archives. Level-1 and level-2 archives are optional in your schedule, but level-0 archives are not. At the very least, you must create level-0 archives to perform the administrative tasks in "Synchronize Administrative Tasks with Archives" on page 5-10.

When you are using ON-Archive, you could also want to archive dbspaces at different times, rather than always archiving all the data managed with Universal Server.

Figure 5-1 on page 5-21 shows three very different archive schedules, ranging from one that creates archives very frequently to one that does not.

Figure 5-1
Examples of Archive Schedules

Level	Daily Schedule	Weekly Schedule	Monthly Schedule
Level-0	Every night	Sundays	Once a month
Level-1	At lunch	Wednesdays	Once a week
Level-2	Every two hours	Mondays, Tuesdays, Thursdays, Fridays	Once a day

Your site can adopt a schedule that looks like one of the preceding schedules, or a modified version of them. The schedule you develop depends on how much time you want to devote to making archives, how much time you can devote to a restore, the number of available tape drives, and the availability of an operator for making archives.

Determining Your Priorities for the Schedule

Each of the following considerations affect the archive schedule you create for your environment:

- Do you need to minimize the time for a restore?
- Do you need to minimize the time to create an archive?
- Do you need to create archives while Universal Server operates in on-line mode?
- Do you need to use the same tape drive to create archives and back up logical-log files?
- Is the operator periodically unavailable?
- When you are using ON-Archive, are some tables more volatile than others, and could require more frequent archiving.

Estimating the Time Required for an Archive

You must consider several variables when you estimate the time it takes to perform an archive. Each of the following items has an impact on the time needed to complete an archive:

- Overall speed of the tape device, including operating-system overhead
- Level of the archive
- Size of the archive
- Amount and type of database activity during the archive
- Amount and type of database activity in the period after the last archive
- Alertness of the operator to tape-changing demands
- Whether you archive different dbspaces concurrently, using multiple tape drives.

For the best approach to estimating the time needed to complete an archive, create an archive and try to gauge the time for subsequent archives using the first archive as a basis for comparison.

Minimizing Archive Size

The size of a full-system level-0 archive is the sum of all in-use data. As mentioned in "Full-System Archives and Dbspace-Set Archives" on page 5-4, you can archive dbspaces in sets. This option enables you to archive the entire system in smaller units and in stages that you can often more easily manage than a full-system archive performed in a single operation.

The size of a level-1 archive is a function of the time and amount of update activity after your level-0 archive. The more often you create level-0 archives, and the less updating between archives, the smaller each level-1 archive can be. Level-2 two archives can also remain smaller when level-1 archives are more frequent and require less updating between them.

Minimizing the Time for an Archive

When multiple tape drives exist, you can reduce the duration of an archive by archiving different sets of dbspaces to different tape drives in parallel—that is, at the same time.

When only one tape drive exists, you can still reduce the duration of individual archives by archiving sets of dbspaces in stages over several days, as described in the preceding section.

To reduce the duration of an archive, reduce the number of data pages that you must archive. To reduce the number of data pages that you must archive, manage space for temporary tables in one of the following ways:

- Create a temporary dbspace, as described in the *INFORMIX-Universal Server Administrator's Guide*, and store your temporary tables there. An archive ignores any tables stored in a temporary dbspace.
- Create a normal dbspace where you store all temporary tables; then, drop the dbspace before you create an archive.
- Drop all temporary tables before you archive the dbspaces where they reside.

You only need to do a level-0 archive after creating a dbspace when you plan to restore it. When you do not create a level-0 archive for the dbspace and a critical media failure occurs, one of the following things happens:

- The archive marks the dbspace as disabled and the chunks go down.
- You recover the dbspace when you replay the logs because the archive logs the statement that created the dbspace.

When the dbspace remains in a disabled state, you can drop the dbspace and re-create it.

How to Minimize the Time for a Restore

The following list shows those thing required to perform a restore:

- Size and number of archives

  The minimum number of archives needed to restore is one level-0 archive. The maximum number is three, one of each archive level.

- Amount of data for restoration
- Size and number of logical-log files after the last archive

  More log files take longer to restore.
- Degree of concurrency

  ON-Archive allows you to perform concurrent physical restores using multiple tape drives.
- Type of restore

  As explained in "Decide on a Cold or a Mixed Restore" on page 7-24, when you perform a full-system restore, you can restore some dbspaces first, while Universal Server is off-line. Then, when Universal Server comes on-line, those dbspaces are available while other dbspaces are being restored. This type of restore increases the availability of some dbspaces but also increases the total restore time. When your logical log becomes full, Universal Server suspends processing until you back it up. This means either that you must abort the archive or processing can remain suspended until you complete the archive and you back up the logical-log files.

Consider the following strategy to minimize the time needed to restore an Universal Server database server:

- Create a level-0 archive as often as is practicable, perhaps every three days.
- Create a level-1 archive daily.
- Do not use level-2 archives.

The time required for any possible restore is limited to the time needed to read and process the following data:

- A level-0 archive of the dbspace(s) being restored
- A level-1 archive, representing from one to three days' activity in the dbspace or dbspaces being restored
- Logical-log files, representing less than a day of work in the dbspace or dbspaces being restored

Monitoring Archive History

To monitor dbspace set archives, use "The LIST / DBSPACESET Command" on page 8-52.

Details of an Archive

This section explains what Universal Server does during an archive. It is not necessary to understand this section to perform an archive; it is provided only as background information.

The steps involved in the process are described in the following list:

- ON-Archive connects and requests an archive.
- ON-Archive readies the device.
- Universal Server prepares to create an archive.
- Universal Server builds and sends archive data.
- ON-Archive writes data.
- ON-Archive and Universal Server commit the archive.

ON-Archive Connects and Requests an Archive

ON-Archive defines a list of dbspaces and blobspaces for archiving and sends it to Universal Server. The list defines the order you use to archive the dbspaces, with the following exceptions: When the rootdbs is among the dbspaces, it gets archived first, and when any of the spaces are blobspaces, they get archived before any of the dbspaces. Universal Server blocks blobpage allocation for each blob until the blobspace is archived. Universal Server releases the block as soon as the blobspace is archived; the blobspace is not blocked until the end of the archive.

Once ON-Archive has completed the connection, it reads archive data generated with Universal Server until no more data exists for archiving.

ON-Archive Readies the Device

When the archive is attended (you are using **onarchive**), you get prompted to mount a tape on the tape device.

When the archive is unattended (you are using **onautovop**), **onautovop** assumes that the tape resides on the device.

Universal Server Prepares to Create an Archive

When Universal Server receives a request for an archive, it performs several tasks:

1. When the archive request includes a blobspace or dbspace that is disabled, Universal Server returns an error and aborts the archive.
2. Universal Server compares the specified archive level with the information in the archive reserved page.

   When Universal Server cannot find a record of a previous archive on the reserved page, the only valid archive level is a level-0 archive. Otherwise, any archive level is valid.

   (A level-0 archive to **/dev/null** registers as a valid archive. Therefore, Universal Server permits you to create a level-1 archive on a tape device when your only level-0 archive was created when the archive device was **/dev/null**. Because of the problems this could create when you need a data restore, avoid this situation.)
3. Universal Server temporarily freezes the status of used logical-log files and checks the total amount of free log space. When free space is less than half of one log file, Universal Server refuses the archive request and recommends that you back up the logical-log files.
4. Universal Server synchronizes with other archiving processes to guarantee that no two archiving processes are simultaneously archiving the same dbspace or blobspace.
5. Universal Server initiates a checkpoint (called the *archive checkpoint*). The checkpoint marks the beginning of the archive.

Universal Server uses a timestamp to determine the order of the pages that get archived. Any pages that are created later than the archive checkpoint are not archived; a page that is modified after the archive checkpoint archives its before-image from the physical log rather than from the modified page.

For example, assume the checkpoint occurs at 3401. (Timestamps are not based on system time.) For a level-0 archive, you must archive all pages containing timestamps less than 3401. As Universal Server reads through disk pages during the archive, pages with timestamps greater than 3401 are ignored. Universal Server relies on the logical-log files to contain records of modifications that occur after 3401.

The address of the most-recently written record in the current logical-log file is also noted during the checkpoint. This record becomes the last record from the logical-log file that is copied as part of this Universal Server archive.

It is likely that some transactions are ongoing during an on-line archive procedure. The restore procedure describes how transactions that span the archive tape and the logical log are rolled back during a data restore, when necessary.

6. Universal Server reads archive history from the archive reserved page, saving the timestamp of the previous archive to set the criteria for determining the correct pages to archive in non-level-0 archives.
7. When the archive is a full-system archive, all the logical-log files that contain log records from open transactions are marked so that they are not freed until they are archived.
8. Universal Server builds a list of free pages within each chunk scheduled for archiving; the unused pages (and the pages devoted to the logical or physical log) at the time of the archive checkpoint are not archived. This information is contained in entries on the dbspace chunk free-list pages and the blobspace free-map page.
9. Universal Server creates a temporary table for each dbspace being archived for storing before-images from the physical log. These temporary tables are stored in the same spaces designated for other temporary tables. (Refer to the *INFORMIX-Universal Server Administrator's Guide* for information on temporary tables.) When Universal Server runs out of space where it can create the temporary tables, it aborts the archive.

Once all of these tasks are performed, Universal Server starts an internal archive thread that is responsible for generating archive data.

Universal Server Builds and Sends Data

This section describes the order of information that Universal Server sends to ON-Archive, and the special actions Universal Server takes to ensure that the archive is complete and efficient.

The Order of the Archive Data

The first information Universal Server sends to ON-Archive is a control page that contains information about the archive, including the following items:

- List of spaces included in the archive
- Archive level
- Archive timestamp
- Logging information

Following the control page, Universal Server sends data in the following order:

1. When the archive includes the root dbspace, then a section including the reserve pages from the root dbspace is added to the archive.
2. When the request is for a full-system archive, a snapshot of the logical-log files that includes open transactions at the time of the archive checkpoint is added to the archive.
3. When the archive contains any blobspaces, they are included next. Archiving blobspaces early allows blobpage allocation to resume as soon as possible because the allocation is blocked at the start of the archive until the blobspace has been archived. Only the used portion of a blobpage is archived, not the whole page.
4. Following the blobspaces, the dbspaces are archived in no particular order.
5. The introductory tape-control page of each dbspace and blobspace section contains a mapping of the chunks contained within the space being backed up.

6. When a dbspace or blobspace has been archived, the pages from the temporary table used to store before-images from the physical log are appended.
7. When Universal Server reaches the last page of the last chunk, the disk-reading portion of the archive procedure is complete. Universal Server sends a trailer page to ON-Archive, marking the end of the archive data.

How the Data Is Built

Universal Server does several things while building the archive data for ON-Archive.

No Waiting for ON-Archive

Universal Server does not wait while ON-Archive writes archive data to the backup media. Universal Server switches to other threads and continues with other processing during an archive. When ON-Archive is ready for more data, Universal Server resumes sending it.

Use of the Physical Log

When a page is updated while an archive is in progress, the archive process retrieves the before-image of the page from the physical-log file to capture the state of the page at the time of the archive checkpoint. Periodically, Universal Server empties the physical log of pages that are no longer needed for fast recovery. When this occurs during an archive, any before-images that are needed for the archive are written to a temporary table. Once a before-image is written to the archive, Universal Server removes it from the temporary table. See the *INFORMIX-Universal Server Administrator's Guide* for a detailed description of physical logging.

Block Blobpage Allocation Until the Chunk Is Archived

Since blobpages do not pass through shared memory, the strategy of archiving from the physical log is insufficient in itself. So, Universal Server must prevent clients from overwriting blobspace blobpages before you archive them.

To accomplish this, Universal Server blocks allocation of blobpages in each blobspace chunk until all used blobpages in the chunk are archived. As soon as the chunk is archived, blobpage allocation in that chunk resumes.

One implication of this implementation is that during an on-line archive, you cannot insert blobs into a blobspace until the blobspace chunk has been archived.

Mirror Chunks Are Not Read Directly

Mirror chunks are not explicitly read for archiving. Pages within a mirror chunk are archived only when Universal Server cannot read the page from the primary chunk.

Apply Archive Criteria

As Universal Server reads each disk page, it applies a set of criteria that determines when you archive a disk page. Each page that meets the following criteria is included in the archive:

- The page has been allocated.

  Universal Server uses the list of free pages that it created at the start of the archive to determine those pages it allocates.

- The page is needed for this archive level.

  A level-0 archive requires Universal Server to archive all used disk pages containing a timestamp less than the begin-archive checkpoint timestamp.

  A level-1 archive directs Universal Server to archive all disk pages containing a timestamp that is less than the archive-checkpoint timestamp but greater than the timestamp associated with the most recent level-0 archive.

  A level-2 archive directs Universal Server to archive all disk pages containing a timestamp that is less than the archive-checkpoint timestamp but greater than the timestamp associated with the most recent level-1 archive.

ON-Archive Writes Archive Data

When ON-Archive starts a new volume in a volume set, it puts a volume header onto the volume. For tapes, this means writing a 512-byte record to the tape; for disk volumes, it means writing an 80-byte file on disk. You do not write volume headers to the save-set files of disk volumes.

When ON-Archive starts a new save set in a volume, it puts a save-set header onto the volume and reserves enough space to write a save-set trailer when the volume runs out, or the save set ends. The save-set headers and trailers each use one 512-byte record.

Each file in a save set (each dbspace and each individual logical-log file is a file) in turn gets a 512-byte record as a header and trailer.

What Is a Block?

ON-Archive writes data to volumes in *blocks*. Blocks are made up of 512-byte records. Each block has a header telling how big the block is (although ordinarily all blocks are the same size). For each block of data ON-Archive writes to a volume, it calculates if the block fits on the volume. When a volume lacks sufficient space for the block, for tape volumes, ON-Archive closes the save set on the current volume and uses the next volume to write the block. For disk volumes that lack sufficient space for a block, ON-Archive writes out data in groups of the smallest possible block size. (See "How Small Can a Block Be?" on page 5-32.)

How Big Is a Block?

The maximum size of a block for a request is specified in one of the following two places:

- The BLOCKSIZE qualifier in the request for the archive or logical-log backup (for tape volumes only)
- The BLOCKSIZE qualifier in the **oper_deflt.arc** file

The qualifier in the command line overrides the qualifier in the **oper_deflt.arc** file.

Usually you do not need to change the size of the blocks ON-Archive uses. The default size is the maximum size, and that is fine for most purposes.

How Small Can a Block Be?

When the amount of space on a disk volume is defined (using MAX_SPACE) as very small, ON-Archive writes out data in the smallest block of data it can—Universal Server page size *3, plus a 512-byte block header—until the maximum space is reached.

ON-Archive and Universal Server Commit the Archive

When ON-Archive has received all the archive data, ON-Archive notifies Universal Server and indicates when you must commit or abort the archive. Committing a level-0 archive backup has the following implications:

- Newly mirrored dbspaces and blobspaces become available.
- When the archive includes the root dbspace, newly added log files become available.
- The archive becomes available for use during a restore.
- After a level 0 archive, when any database logging changes are pending, the time of the database-logging change request is compared to the last level-0 archive for each dbspace and blobspace that makes up the database. When you archive all the dbspaces and blobspaces after you make the database logging change request, the logging change takes effect and database access is granted.

When the archive is a full-system archive, Universal Server commits the archive by storing the history of the archive in the archive reserve pages. This information is used with subsequent archive increments.

When the archive is terminated, the temporary tables used for physical-log pages are dropped.

Chapter

6

Backing Up the Logical Log

This chapter describes how back up your logical-log files. It discusses the following topics:

- Considerations before you back up the logical-log files
- The BACKUP command
- Types of logical-log backups
- What to do when a logical-log backup fails
- Details of what happens during a logical-log backup

The *INFORMIX-Universal Server Administrator's Guide* thoroughly describes the logical log and the logical-log record types. It also describes what transaction logging is, how large the logical log could be, where it could reside, which database server activities get logged, and so on.

Before You Back Up the Logical-Log Files

Before you back up the logical-log files, you need to understand the following issues:

- Whether you need to back up the logical-log files
- When you need to back up the logical-log files

When you decide you need to back up the logical-log files, you must decide on the type of backup you want to perform—automatic or continuous.

The following sections explain these issues.

Do You Need to Back Up the Logical Log Files?

When you specify logging for your databases, Universal Server records transactions that occur between archives in the *logical log*, which consists of a finite number of *logical-log files* on disk. Universal Server continually needs to write new log records but also retains the log records it has already written in case you need to restore those transactions. To retain the records in the logical log, yet allow Universal Server to continue writing new log records in a finite amount of space, you must copy full log files to a safe place on disk or tape in order to free them. See "What Is a Logical-Log Backup?" on page 1-4 for more information on logical-log backups.

When You Do Not Use Logging

You must know that even when you do not use logging for any of your databases, you can still perform log backups. These backups are very small because they contain only administrative information such as checkpoint records and additions and deletions of chunks. The process of backing up these logical-log files allows you to perform warm restores even though you do not use logging for any of your databases.

Blobspace Blobs and Logical-Log Files

You must keep the following two points in mind when you use blob data in a database that uses transaction logging:

- To ensure timely reuse of blobpages, you need to back up logical-log files. When users delete blobs in blobspaces, the blobpages are not freed for reuse until you free the log file containing the delete records. To free the log file, you must back it up.
- When you must back up an unavailable blobspace, ON-Archive skips it, making it impossible to recover the blob when that becomes necessary. (However, blobpages from deleted blobs do become free when the blobspace again becomes available, even though the blob was not backed up.)

In addition, regardless of whether the database uses transaction logging, when you create a blobspace or add a chunk to a blobspace, the blobspace or new chunk remains unavailable for use until the log file that records the event is not the current log file. See the *INFORMIX-Universal Server Administrator's Guide* for information on switching log files.

When You Do Not Need to Recover

When you decide that you do not need to recover transactions or administrative database activities between archives, you can set the Universal Server configurations parameter LTAPEDEV to **/dev/null**. When you do this, it has the following implications:

- You can restore only the data managed with your Universal Server database server up to the point of your most recent archive and any previously backed-up logical-log files.
- When you recover, you must always perform a full-system restore. (See "A Full-System Restore" on page 7-5.) You cannot perform partial restores or restore when Universal Server operates in on-line mode.

When you set LTAPEDEV to **/dev/null**, Universal Server marks a logical-log file as backed up (status B) as soon as it becomes full. When you close the last open transaction in the log, the log file is marked free (status F). Universal Server can then reuse that log file without waiting for you to back it up. As a result, you cannot preserve the logical-log records.

Other Universal Server mechanisms that use the logical log, like fast recovery and rolling back transactions, remain fully functional when you use **/dev/null** as your log-file backup device. See the *INFORMIX-Universal Server Administrator's Guide* for a description of Universal Server fast recovery. See the ROLLBACK WORK statement in the *Informix Guide to SQL: Syntax* for information about rolling back transactions.

Warning: *Although ON-Archive does not read the configuration parameters in the $ONCONFIG file, when you set* ***LTAPEDEV*** *to* ***/dev/null****, Universal Server automatically marks full logical-log files as backed up before you can actually back them up.*

When Must You Back Up Logical-Log Files?

You must attempt to back up each logical-log file as soon as it fills. You can tell when a log file is ready for backing up because it has a *used* status. For more information on monitoring the status of logical-log files, see the *INFORMIX-Universal Server Administrator's Guide*.

When you do not want to monitor the log files and start backups when the log files become full, you can use continuous backups.

Automatic and Continuous Backups

To back up all the full log files, you can start an *automatic* backup. An automatic back up automatically backs up all the full logical-log files and then stops at the current log file.

You can also start a *continuous* backup in which Universal Server automatically backs up each logical-log file as it becomes full. When you perform continuous logical-log file backups, you can never lose more than a partial log file, even in the worst-case media failure when a chunk containing logical-log files fails.

With continuous backups you are not required to remember to back up the log files, but someone must ensure that media always remains available for the backup process. Also, you must dedicate the backup device and a terminal to the backup process.

Choosing a Device for Your Logical-Log Backup

Consider the following issues when you choose a logical-log backup device:

- When the logical-log device differs from the archive device, you can plan your backups without considering the competing needs of the archive schedule.
- When you specify **/dev/null** as the logical-log backup device in the ONCONFIG parameter LTAPEDEV, you avoid having to mount and maintain backup tapes. However, you can only recover Universal Server data up to the point of your most-recent archive tape. You cannot restore work done after the archive. Also, you cannot perform a warm restore, and that means that you must always do a full-system restore, instead of restoring only some dbspaces.
- When your tape device is slow, the logical log could fill up faster than Universal Server can copy it to tape. In this case, you can perform the backup to disk and then copy the disk backup to tape.
- When you use a remote logical tape device to perform a continuous backup of logs, and a high volume of log activity exists, the logs can fill up faster than Universal Server can back up the data. Under those conditions, those logs are not freed or marked as backed up. When this situation occurs, you can perform backups to a local tape device so that you can free a log as soon as you back it up.

See "What Devices Does ON-Archive Use?" on page 2-17 for a general description of ON-Archive devices. See "DEVICE" on page 3-8 for more specific information on devices that ON-Archive can use.

Saving Logical-Log Backup Data

The nature of logical-log data makes backing up logical-log files different from creating archives. For this reason, consider taking the following steps:

- Make a copy of all logical-log volumes.

  When one of your volumes with archive data becomes inaccessible, you can always restore from an older archive, when one exists. When a volume containing logical-log files fails, however, you cannot roll forward the transactions from those log files or any subsequent log files.

  See the COPIES qualifier under "Archive and Backup Qualifiers" on page 8-70 and "The COPY / VSET Command" on page 8-24 for information that describes how to create copies of archives and backups.

- Use the EXPIRY_DATE qualifier with the BACKUP command to set expiration dates for save sets containing logical-log files. Setting an expiration date far enough in the future prevents logical-log-backup data from being overwritten inadvertently. See the EXPIRY_DATE qualifier under "Archive and Backup Qualifiers" on page 8-70 for more information. Also see "When Can Volumes Be Reused?" on page 4-45.

Keep copies of your logical-log files until you no longer need them to complement a restore from an archive tape.

Backing Up the Logical-Log Files

Before you back up the logical-log files, you need to understand and complete the tasks described in the following sections:

- Start **oncatlgr**.

  See "Starting the Cataloger" on page 4-5.
- Start **onarchive**.

  See "Starting onarchive" on page 4-7.
- Define a volume set (vset) and volumes to store the backup data.

  See "Using Volume Sets and Volumes" on page 4-31.
- Understand how to create and execute an ON-Archive command.

  See "Creating and Executing Requests" on page 4-26.

Under normal circumstances, you create a logical-log backup using the BACKUP command. See "Performing Emergency Logical-Log File Backups" on page 6-13 for information on how to back up the logical-log files when the ON-Archive catalog is not available.

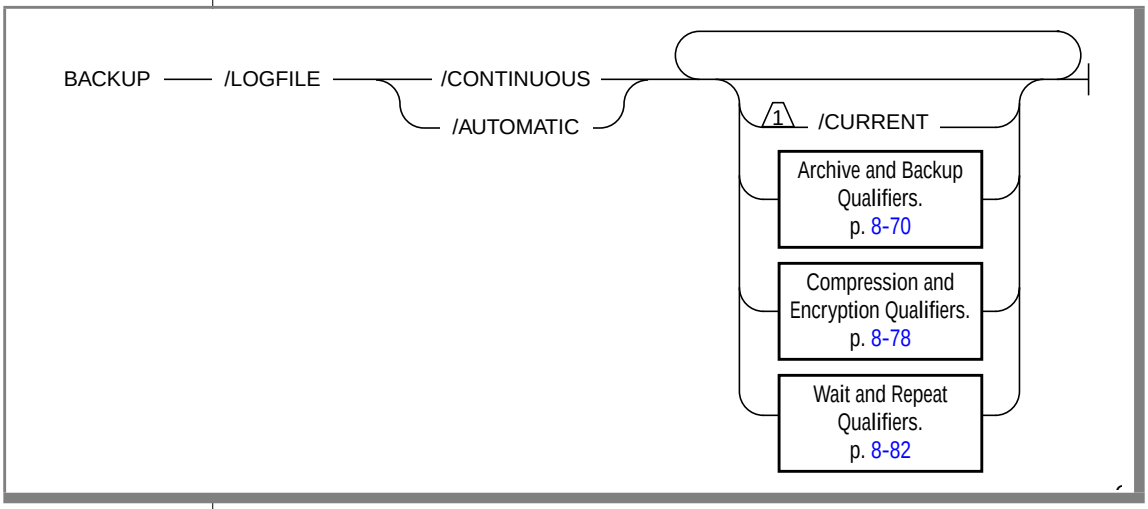
The BACKUP Command

The BACKUP command backs up logical-log files. You must back up the logical-log files independently of creating an archive. The following restrictions apply to the BACKUP command:

- When ON-Archive operates in OPERATOR mode, only users **root** and **informix** can create requests to back up log files.
- When ON-Archive operates in GROUP mode, only users in the **super_archive** group can create requests to back up log files.

When you restore logical-log files, you must possess the necessary privileges to read all of the save sets.

Once you create the requests, restrictions also apply to their execution. See "The EXECUTE Command" on page 4-27 and "The EXECUTE / VSET Command" on page 8-33.



BACKUPLOGFILE specifies that ON-Archive back up logical-log files.

CONTINUOUS specifies that ON-Archive continually back up logical-log files as they become full. See "Performing a Continuous Backup of the Logical-Log File" on page 6-11 for more information about continuous logical-log backups. See "Stopping a Continuous Backup" on page 6-12 for instructions on stopping continuous log backups.

AUTOMATIC specifies that ON-Archive backs up every full logical-log files at the time the request executes.

CURRENT specifies that ON-Archive back up the current, unfilled logical-log file, and switch the current log file to the next log file. When you do not specify CURRENT, the BACKUP command does not back up the current log file.

BACKUP Examples

ON-Archive provides for the following four types of logical-log backups:

- One-time (automatic) backup
- Continuous backup
- Emergency backup when Universal Server processing stops
- Salvaging log files during recovery

Backing Up All Full Logical-Log Files

To make a one-time backup of the full logical-log files, you must use the AUTOMATIC qualifier. The AUTOMATIC qualifier backs up all the full log files at the time you execute the request.

```
Onarchive> BACKUP/LOGFILE/AUTOMATIC/VSET=logvset
```

All the qualifiers (except LEVEL and DBSPACESET) that are valid when you create an archive are also valid for the BACKUP command. See "Archive and Backup Qualifiers" on page 8-70 for descriptions of these qualifiers.

Backing Up the Current Log File

To back up the current logical-log file and switch to the next logical-log file, use the CURRENT qualifier, as in the following example:

```
Onarchive> BACKUP/LOGFILE/AUTOMATIC/CURRENT/VSET=logvset
```

Performing a Continuous Backup of the Logical-Log File

Use the CONTINUOUS qualifier with the BACKUP command to start a continuous backup of the logical-log files. The following example illustrates the use of the CONTINUOUS qualifier:

```
Onarchive> BACKUP/LOGFILE/CONTINUOUS/VSET=logvset
```

When you specify a continuous backup of the logical-log files, ON-Archive backs up log files as they become full and eligible for backup. During a local continuous backup, **onarchive** writes N-2 logs into a save set, where N represents the number of active logical-log files. When each save set is completely written, **onarchive** tells Universal Server to free the log files that were backed up. When you use a remote device for the continuous backup, however, **onarchive** writes each save set to a separate volume.

Important: *In the on-line message log, you see a message such as the following one each time you back up a file:*

```
14:13:05 Logical Log 12 - Backup Started
```

When you request verification, the process performing the backup, either ***onarchive*** *or* ***onautovop****, does not "commit" the backup and tell Universal Server to free the log file until the entire save set is written and verified. At the time the "commit" finishes, you see the following message in the on-line log:*

```
14:13:21 Logical Log 12 - Backup Completed
```

You can use the ***onstat -l*** *command to verify that the database server has marked the logical- log file as free for subsequent use. See the "INFORMIX-Universal Server Administrator's Guide" for more information on how to use the* ***onstat*** *utility.*

Using onarchive or onautovop

When **onarchive** starts continuous log-file backups, the window or terminal from which you execute the command is dedicated to the backups.

When **onautovop** starts continuous log-file backups, that **onautovop** process is dedicated to executing that request.

Stopping a Continuous Backup

The method of stopping a continuous backup of the logical-log files depends on whether you use **onautovop** or **onarchive** to back up the log files.

When you use **onautovop**, you must stop the **onautovop** process. (See "Using Unattended Operations" on page 4-50 and "The stop_autovop Script" on page 9-33 for information on stopping **onautovop**.)

When you use **onarchive**, you use the Interrupt key, ^C (CTRL-C), to stop continuous logical-log file backups.

When you press the Interrupt key while Universal Server backs up a logical-log file to a local device, all logs that were completely backed up before the interrupt are captured on the tape and are marked as backed up with Universal Server.

When you press the Interrupt key while Universal Server waits for a log file to fill (and does not engage in backing up any logical-log files), all logs that were backed up before the interrupt are on the tape and marked as backed up with Universal Server.

When you press the Interrupt key while Universal Server performs a continuous backup to a remote device, any log files that were backed up during this operation can or cannot exist on the tape and are not marked as backed up with Universal Server.

Performing Emergency Logical-Log File Backups

As explained in the *INFORMIX-Universal Server Administrator's Guide*, when your logical-log happens to fill (or, more accurately, reach the degree of fullness specified in the LTXEHWM parameter in the ONCONFIG file), Universal Server activity stops. You cannot use the ON-Archive BACKUP command to back up the logical-log files in this situation because it requires accessing the ON-Archive catalog, and that access is prohibited when the logical log has filled. Consequently, you must use **ondatartr** to back up your log files in this situation. See "Emergency Situations" on page 4-48 for more background information. See the section "The ondatartr Utility" on page 9-12 for the syntax of **ondatartr**.

To perform an emergency logical-log backup, perform the following steps in order:

1. Leave Universal Server in on-line mode. Also leave **oncatlgr** and **onarchive** or **onautovop** running.
2. Prepare media for emergency log backup.
3. Use **ondatartr** to back up the log files. After you use **ondatartr** to back up the log files, the database server continues normal execution without any further action required.
4. Catalog the volume created by **ondatartr** into the ON-Archive catalog.

5. Copy the save set to the vset where you normally store your backed up log files.
6. Remove the files from the emergency vset.

The following sections explain each of these steps.

Leave Universal Server in On-line Mode

Do not take Universal Server off-line to perform emergency log backups.

Prepare Media for Emergency Log Backup

It does not matter what kind of media (disk or tape) you use for emergency log backups because you eventually copy the backed up logs to the vset where other logical-log files reside. When you previously define emergency volumes, as described in "Defining an Emergency Volume Set and Volume" on page 4-48, you must prepare the same type of media (disk or tape) used to define the emergency vset.

When you use a tape volume for emergency log backups, use a tape with **ondatartr** that has not been initialized as an ON-Archive volume.

When you use a disk volume for emergency log backups, you require no action as to prepare the media. When you run **ondatartr**, you specify the directory that was created as the emergency volume in "Defining an Emergency Volume Set and Volume" on page 4-48.

When you did not previously define an emergency log volume, you need to create a new directory for the disk volume. Create the new volume in a directory used as a physical device in the ON-Archive configuration file, **config.arc**. For example, the following command creates a directory in the **/archive** directory, for use as a physical device:

```
% mkdir /archive/emergency_log
```

The permissions on the file require that the user who runs **ondatartr** (**informix** or **root**) can access the file.

Use ondatartr to Back Up the Log Files

Start **ondatartr** and back up the log files. Figure 6-1 shows the interaction of **ondatartr** during a log backup, using the directory **/archive/em_logs** as a volume.

```
% ondatartr
ONDATARTR>
BACKUP/LOGFILE/DISK=(/archive/em_logs)/MAX_SPACE=4000
ONARCHIVE: Media Initialization Function

Disk Information:

        Vset:       ONDATARTRLOG
        Volume  :   0001
        Virtual:    /archive/em_logs

Initialize this volume?

(y/n): y
09:43:53 Logical Log 253 - Backup Started
Save set 1103 being written to vset ONDATARTRLOG.
Logfile LF00000253 backed up.

09:44:33 Logical Log 254 - Backup Started
Logfile LF00000254 backed up.

09:45:15 Logical Log 255 - Backup Started
Logfile LF00000255 backed up.
09:46:15 Logical Log 253 - Backup Completed
09:46:15 Logical Log 254 - Backup Completed
09:46:23 Logical Log 255 - Backup Completed
ONDATARTR> EXIT
```

Figure 6-1
Sample ondatartr Log Backup Interaction

The save set id (**1103**, in Figure 6-1) is the process id of the **ondatartr** process.

Although the logical-log files are backed up and Universal Server processing can continue, the backed-up log files are not yet part of the ON-Archive catalog and you cannot use them during a restore. Also, when you need to perform a restore, the log files used in a logical restore must all come from the same vset. So, to perform a logical restore using these log files, you must take the following actions:

- Use the CATALOG command to enter the volume and save set created by **ondatartr** into the ON-Archive catalog.
- Use the COPY command to copy the save set to the vset where you normally store your backed-up log files.

After that, Informix recommends that you remove the files from the emergency vset.

Catalog the Emergency Volume

Cataloging volumes is the process of putting information about a save set on a volume into the ON-Archive catalog. You use the CATALOG command to catalog the emergency volume, as described in "The CATALOG Command" on page 8-21.

You must define the volumes in the catalog before you catalog the emergency volume or volumes created by **ondatartr**. When you do not previously define a volume and vset, as described in "Defining an Emergency Volume Set and Volume" on page 4-48, do so now. When you use a disk volume with **ondatartr**, define the volume using the pathname used in **ondatartr**.

To catalog tape volumes created by **ondatartr**, mount the first tape with the **ondatartr** save set on the device for the save set you specified when you defined the ONDATARTRLOG vset in ON-Archive.

To catalog disk volumes, make sure the directory is readable.

Then, use the CATALOG command, as shown in the following example:

```
Onarchive> CATALOG/VSET=ONDATARTRLOG/VOLUME=1
```

When **ondatartr** created more than one volume, repeat the command for each of the volumes created by **ondatartr**.

ON-Archive tells you the request id it assigns to the save set when cataloged:

```
Save set 1103 cataloged as request 144.
```

Copy the Volumes to the Regular Log Volume Set

After you catalog the **ondatartr** volumes, copy the save set of the log files to the vset where you ordinarily keep your logical-log files, as shown in the following example:

```
Onarchive> COPY/VSET=ONDATARTRLOG/DESTINATION=logvset
```

This command ensures that you can use the log files in a restore, when necessary.

Clear the Volume Set

Once you copy the volumes in the ONDATARTRLOGS vset to the vset where you normally store your logical-log files, you must use the REMOVE/REQUEST command to clear out the ONDATARTRLOG vset in case you need to use it again.

```
Onarchive> REMOVE/REQUEST=144/VSET=ONDATARTRLOG
```

The request id is the one reported by the CATALOG command.

When you use a disk volume, you can remove the save-set files using the operating system.

When the Logical-Log Backup Cannot Complete

When you back up a logical-log file and a failure occurs that stops the backup before back up completes, you must consider taking either of the following actions:

- Perform another log-file backup (when you do not perform a restore as a result of the problem that caused the log-file backup to fail)
- *Salvage* the log files on disk (when you do perform a restore as a result of the problem that caused the log-file backup to fail).

The following paragraphs explain these actions.

Back Up the Log Files Again

When a logical-log file backup fails, the next logical-log file backup session (when ever that could be) begins with the logical-log file that was being backed up when the failure occurred. The new logical-log backup backs up the entire log file.

When you must restore from these backup tapes (one of which contains a partial log file, and one of which contains a complete copy of the same log file), you can roll forward all the backup tapes, including the one with the partial log file.

Universal Server handles the partial tapes transparently, but when you are curious about how it works, you can read about it in "Details of a Logical-Log File Backup" on page 6-19.

Salvaging the Log Files

When you perform a cold restore (see "A Cold Restore" on page 7-7), you can perform *salvaging* of the log files on disk. Salvaging backs up to tape any logical-log files on disk that are not marked as backed up. When you choose to salvage the logs, it is the first thing the restore procedure does. The log file whose backup was interrupted was not marked as backed up when the failure occurred, so you can salvage it along with any other log files that were not backed up.

Details of a Logical-Log File Backup

This section describes the steps that the backup client, **onarchive** or **onautovop,** and Universal Server perform to back up the logical log to tape. You do not need to understand this section to back up logical-log files; it is here for your information only.

The following list describes the steps:

- The client connects to Universal Server and requests a log-file backup.
- The client readies the device and tape.
- Universal Server prepares to back up log data.
- Universal Server builds and sends logical-log file data.
- The client writes the logical-log backup data.
- The client and Universal Server commit the back up.

The following sections explain these steps in detail.

Client Connects and Requests a Backup

The client connects to Universal Server and sends a request to back up the logical-log files.

Client Readies the Device and Tape

When the client is **onarchive** (an attended backup), **onarchive** prompts you to mount a tape on the tape device specified in the configuration file.

When the client is **onautovop** (an unattended backup), the tape is assumed to reside on the device.

When the tape is new, the client writes a tape header (also called a volume header) to the device.

Universal Server Prepares to Back Up a Logical-Log File

When Universal Server receives a request for a log-file backup, it locates the oldest logical-log file that has been used but not backed up (status U). Universal Server also checks to see that no other log backups are occurring.

Next, Universal Server starts an internal thread that is responsible for collecting the log-file data and sending it to the client.

Universal Server Builds and Sends Logical-Log Data

Universal Server builds the data that you must back up and sends it to the client. The client writes the data to the backup device.

Universal Server Sends Blobpages

Universal Server begins by comparing the identification number of the log file it backs up with every blobspace blob. (It actually looks at every blobspace free-map page). Universal Server looks for blobpages that were allocated or marked for deletion during the time this logical-log file was the current log file.

When you must copy blobpages, each blobpage that was allocated or marked for deletion during the time that this log file was current gets sent to the client. Universal Server precedes each blobpage with a blob header and follows it with a blob trailer.

When a blobspace containing blobpages that you must back up is unavailable at the time a backup (or salvaging) of the log files occurs, Universal Server does not wait for the blobspace to become available. It continues the log-file backup without copying the blobpages it needs. Thus, you cannot restore the blob when you roll the logical-log file forward, and you lose the blob during a restore.

Universal Server Sends a Log Header

After you check all blobs, and after you send the required blobpages to the client, Universal Server creates a log header and sends it to the client.

The log header is distinct from the tape header. The log header specifies (among other things) the id number of the logical-log file and the number of pages from the logical-log file that you must copy.

Universal Server Sends Log Records

Following the log header, Universal Server begins sending each page in the logical-log file that it backs up. When some pages in the log file are not used (for example, when you back up a file before it is full), Universal Server does not write the unused pages in the log file to tape.

Universal Server Sends a Log Trailer

After it sends the last page in the log file, Universal Server sends the client a log trailer.

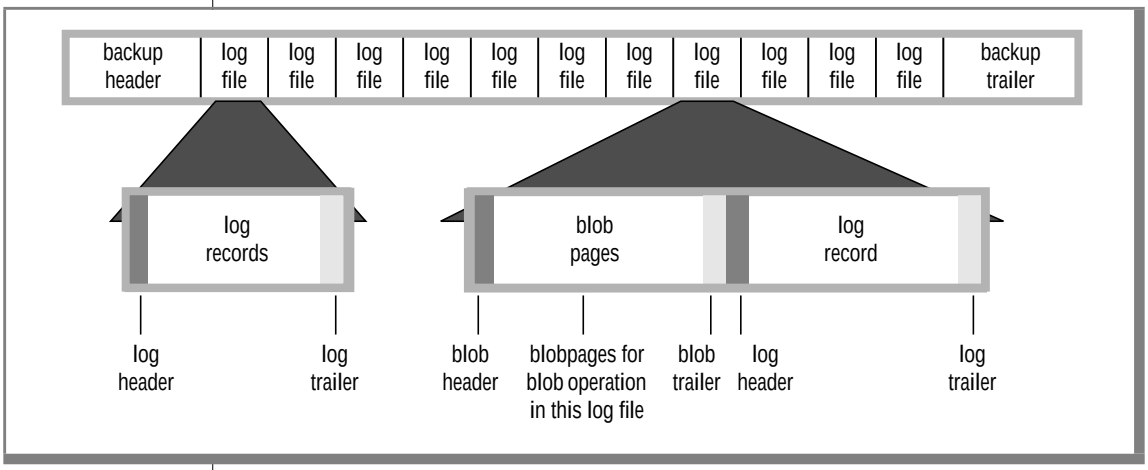
Client Writes Data to the Backup Device

The client writes the logical-log data to the backup device in the same order that it receives it from Universal Server:

1. Blobpages
2. Log file header
3. Log file records
4. Log file trailer

Figure 6-2 illustrates the order of information on the logical-log backup tape.

Figure 6-2
Format of Logical-Log Backup Tape



The Client and Universal Server Commit the Backup

When the client has secured all the log-file backup data, the client notifies Universal Server and indicates whether you must commit or abort the log-file backup. Committing the log-file backup changes the status to backed-up. Aborting the log-file backup leaves the log file in the same state as it was prior to the log-file backup. When it changes the log-file status, Universal Server checks to see if it can free the log file for reuse.

The Client Looks for More Log Files to Back Up

After you successfully back up a logical-log file, the client determines whether you must back up another log file. When you do not need to back up other log files and the user specified that you must back up the current log file, Universal Server switches the current log file to the next log file and backs up the (formerly) current log file. When the log backup continues, it proceeds by repeating the steps outlined in the preceding three sections: "Universal Server Builds and Sends Logical-Log Data" on page 6-20, "Client Writes Data to the Backup Device" on page 6-21, and "The Client and Universal Server Commit the Backup" on page 6-22.

When you do not need to back up more logs and the client performs a continuous log backup, the client goes into a loop where it waits awhile and then again asks Universal Server if you need to back up any full log files. As the log files fill, they are backed up.

Client Writes the Trailer Page

When the entire log backup process finishes, the client writes a backup trailer to indicate the end of the backup session.

When a New Volume Is Needed

When you need more than one tape during the logical-log backup, the client provides you with labelling information for the full tape and prompts you to mount a new tape.

Chapter

7

Restoring Universal Data

This chapter explains how to restore Universal Server data from an archive and a set of backed-up logical-log files. First, the chapter explains what types of restore operations you can perform and what situations require each of them. Next, it describes the RETRIEVE / DBSPACESET and RETRIEVE / LOGFILES commands, that restore data from an archive and from a logical-log backup, respectively. Last, the chapter describes the steps you must follow to restore either all of your Universal Server data or only selected dbspaces.

Before You Restore

An Universal Server restore re-creates Universal Server data that has become inaccessible due to some type of hardware or software failure. For example, any one of the following three conditions could require you to restore your Universal Server data.

- You need to replace a disk that contains Universal Server data.
- A logic error in a program has corrupted a database.
- You need to move your Universal Server data to a new computer.

To restore your Universal Server data up to the time of the failure, you must have an archive of your Universal Server data and the logical-log files that contain all transactions after the archive was created. See "What Is a Universal Server Restore?" on page 1-6 for a description of an Universal Server restore. Chapter 5, "Creating an Archive," explains how to create an archive of your Universal Server data and Chapter 6, "Backing Up the Logical Log," explains how to back up the logical-log files.

The following sections describe the process of restoring your Universal Server data and explain the choices that you need to make before you perform a restore.

Understanding Physical and Logical Restores

When you restore Universal Server data, you restore it in two separate operations—a *physical* restore followed by a *logical* restore. The following list defines these operations:

- Physical restore

  A physical restore restores Universal Server dbspaces or blobspaces by copying them from an archive that you created previously.

- Logical restore

  A logical restore restores database transactions and system changes that occurred *after the archive that was used in the physical restore.* A logical restore restores transactions and system changes by rolling them forward from logical-log files that you backed up after the archive. When logical-log backups are not available, you lose all changes made after the archive. See "Do You Need to Back Up the Logical Log Files?" on page 6-4 to decide whether you need to back up your logical-log files.

A physical restore does not necessarily need to restore all the dbspaces or blobspaces managed by Universal Server. For example, you could need to restore only the dbspaces with chunks that resided on a disk drive that failed. The section "A Warm Restore" on page 7-8 describes how to selectively restore dbspaces.

A logical restore reads the logical-log records for all the Universal Server dbspaces and blobspaces that were active in the log. However, it only applies transactions to those dbspaces and blobspaces that you just physically restored. Universal Server rolls these transactions forward within a single logical restore operation.

Universal Server rolls back any unresolved transactions when you complete the logical restore.

You use two different ON-Archive commands to perform the physical and logical restores. You use the RETRIEVE / DBSPACESET command to perform a physical restore and you use the RETRIEVE / LOGFILE command to perform a logical restore. The use of two commands allows you to perform multiple physical restore operations concurrently (that is, restore multiple dbspaces concurrently) followed by a single logical-restore operation.

Concurrency During a Logical Restore

During a logical restore, Universal Server reads and sorts log records using the tblspace they affect. Universal Server rolls the log records forward for a given tblspace using special recovery threads in the order that they were initially applied. Multiple recovery threads provide some concurrency, that speeds up the logical restore, allowing multiple tables to evolve to a consistent state at approximately the same time. The number of threads used in the logical restore is determined by the ON_RECVRY_THREADS and OFF_RECVRY_THREADS parameters in the ONCONFIG file. The specific parameter you use depends on whether Universal Server operates in on-line or off-line mode during the logical restore. See the *INFORMIX-Universal Server Administrator's Guide* for more information on these parameters.

Some records, like those that signal whether you must commit or roll back a transaction, are not assigned to a particular tblspace. The recovery threads ensure that those records are handled at the end of a transaction.

Choosing the Type of Physical Restore

When you must restore Universal Server data spaces due to a failure that *caused* Universal Server *to go to off-line mode*, you must restore all the data managed with Universal Server. This type of restore is called a *full-system* restore. When the failure did not cause Universal Server to go to off-line mode, you can choose to restore only selected dbspaces and blobspaces—that is, only those dbspaces and blobspaces that were affected by the failure.

The following sections explain these two types of physical-restore operations, and the conditions that determine the type you must perform.

A Full-System Restore

When your Universal Server database server goes to off-line mode because of a disk failure or corrupted data, it means that a *critical dbspace* was damaged. The dbspaces in the following list are considered critical dbspaces:

- The root dbspace
- The dbspace containing the physical log
- A dbspace containing logical-log files

When you need to restore any critical dbspace, you must perform a full system restore to restore all the data managed by your Universal Server database server. You must start a full system restore with a *cold restore*. See "Choosing a Universal Server Mode—Cold, Warm, or Mixed Restore".

Restoring Selected Dbspaces and Blobspaces

When your Universal Server database server *does not* go to off-line mode because of a disk failure or corrupted data, the damage occurred to a noncritical dbspace or blobspace.

When you do not need to restore a critical dbspace, you can restore only those dbspaces and blobspaces that contain a damaged chunk or chunks. When a a media failure occurs in one chunk of a dbspace or blobspace that spans multiple chunks, all active transactions for that dbspace or blobspace must terminate before Universal Server can restore it. You can start a restore operation before you finish the transactions, but you delay that restore until Universal Server verifies that all transactions that were active at the time of the failure have finished.

Choosing a Universal Server Mode—Cold, Warm, or Mixed Restore

When you restore Universal Server data, you must decide whether you must do it while Universal Server operates in off-line mode or on-line mode. This decision is not completely arbitrary, however. It depends in part on the data you are restoring. The following sections explain the factors that determine the Universal Server mode you must use when you perform a restore.

A Cold Restore

A *cold restore* is a restore that you perform while Universal Server operates in off-line mode. It consists of both a physical restore and a logical restore. You must perform a cold restore to restore any critical dbspaces.

As shown in Figure 7-1, you can restore all the dbspaces and blobspaces managed by Universal Server (a full-system restore) with one physical restore and one logical restore.

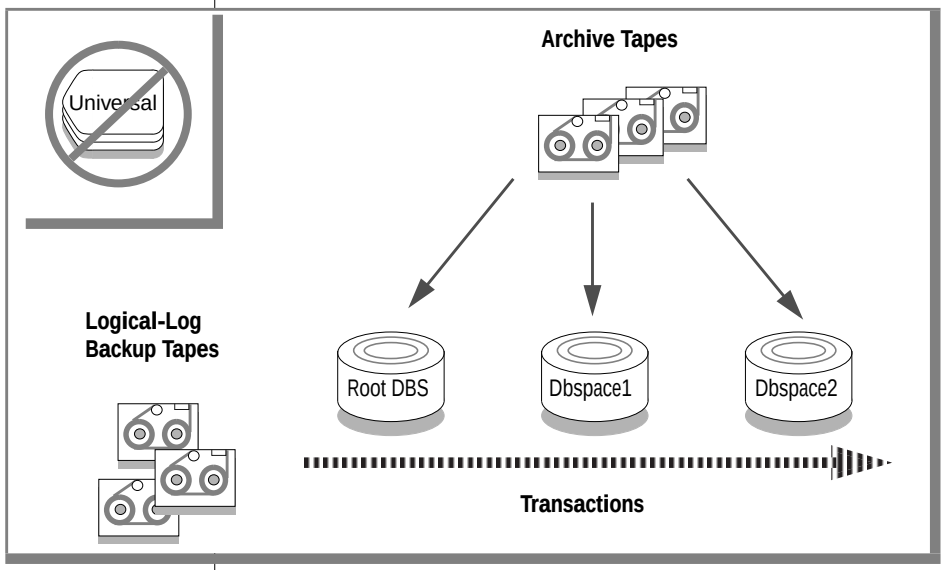


Figure 7-1
A Full-System Cold Restore

Universal Server operates in off-line mode when you begin a cold restore but it goes into recovery mode after you restore the reserved pages. From that point on, it stays in recovery mode until you complete a logical restore (after which it remains in quiescent mode) or after you use the **onmode** utility to place it in another mode.

When you wish, you can perform a cold restore on only some of your dbspaces and restore the remaining dbspaces after you bring Universal Server into on-line mode. Again, however, you must restore any critical dbspaces during the cold restore. See "A Mixed Restore" on page 7-9 for a description of this type of restore.

The logical restore that you perform during a cold restore uses the same space to sort log records that you devote to the logical-log files during normal Universal Server processing.

A Warm Restore

A warm restore restores noncritical dbspaces and blobspaces while Universal Server operates in on-line or quiescent mode. It consists of one or more physical-restore operations (when you are restoring multiple dbspaces or blobspaces concurrently), a logical-log backup, and a logical restore. Figure 7-2 depicts a warm restore.

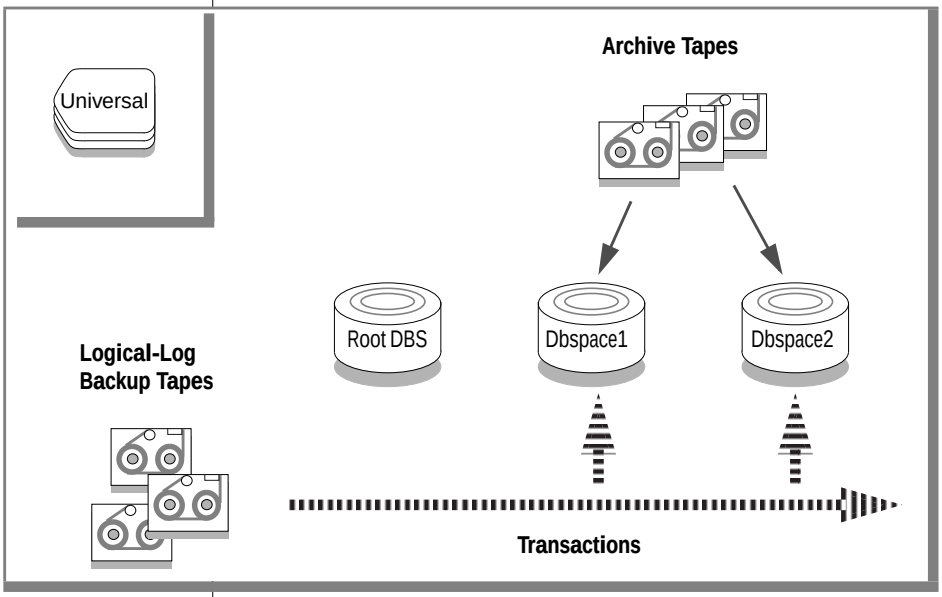


Figure 7-2
A Warm Restore

During a warm restore, Universal Server replays backed-up logical-log files for the dbspaces it restores. To avoid overwriting the current logical log, Universal Server writes the logical-log files to temporary space. Therefore, a warm restore requires enough temporary space to hold the logical log (one set of logical-log files) or the number of log files being replayed, whichever is smaller. See the discussion of DBSPACETEMP in the *INFORMIX-Universal Server Administrator's Guide* for information on how INFORMIX-Universal Server looks for temporary space.

Warning: *Make sure you allot enough temporary space for the logical-log portion of the warm restore; the maximum amount of temporary space that Universal Server needs equals the size of the logical log (the size of all the logical-log files).*

A Mixed Restore

A *mixed restore* is a cold restore followed by a warm restore. A mixed restore restores some dbspaces and blobspaces during a cold restore (Universal Server is off-line) and some dbspaces and blobspaces during a warm restore (Universal Server is on-line). You could do a mixed restore when you perform full-system restore but you need to provide access to a particular table or set of tables as soon as possible. In this case, you perform a cold restore to restore the critical dbspaces and the dbspaces containing the important tables. Figure 7-3 illustrates the cold portion of a mixed restore.

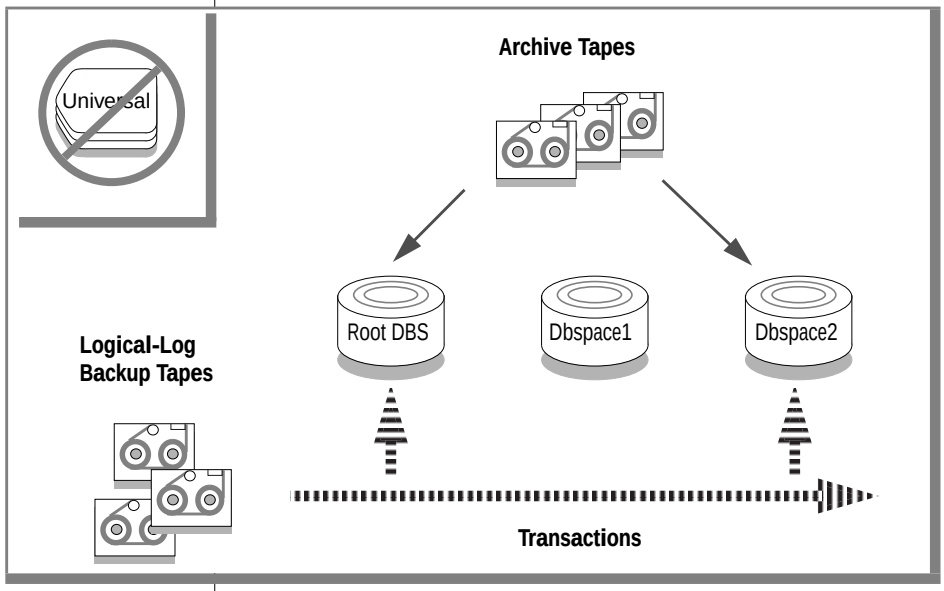


Figure 7-3
The Cold Portion of a Mixed Restore

Following the cold restore, you place Universal Server in on-line mode and perform a warm restore to restore the remaining dbspaces. Figure 7-4 illustrates the warm portion of a mixed restore.

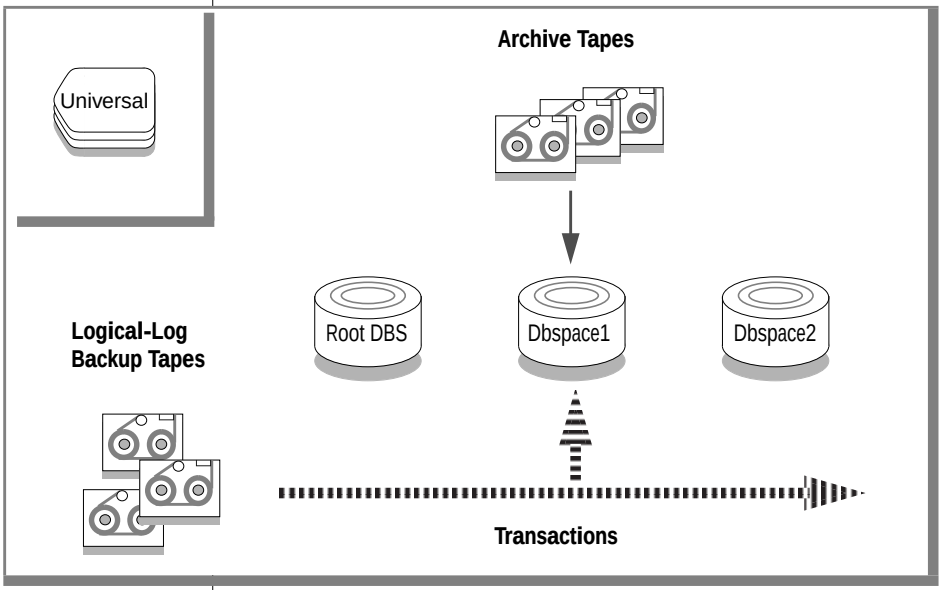


Figure 7-4
The Warm Portion of a Mixed Restore

A cold restore takes less total time to restore all of your Universal Server data than a mixed restore, even though Universal Server is on-line during part of a mixed restore. The reason for this is that a mixed restore requires two logical restores (one for the cold restore and one for the warm restore). A mixed restore, however, requires Universal Server to operate *off-line* for less time than a cold restore.

The dbspaces not restored during the cold restore remain unavailable until after they are restored during a warm restore, even though a failure to a critical dbspace can damage them.

Catalog the Save Set When You Salvage Your Log Files

When you perform a mixed restore using ON-Archive and you salvage your logical-log files during the cold restore, after Universal Server is on-line you must catalog the save set created during the salvage operation. You must then copy the save set to the volume set that contains the rest of your logical-log files before you perform the warm restore. When you do not catalog and copy the salvaged log files, ON-Archive cannot use them during the logical restore, because it can only perform one logical restore and can only use log-file save sets from one volume set for the operation. See "Defining an Emergency Volume Set and Volume" on page 4-48 for more information on cataloging the volume set and volume that you use to salvage the logical-log files.

Restoring Multiple Dbspaces or Blobspaces in Parallel

You can restore multiple dbspaces and blobspaces concurrently when you archive them in different save sets and you possess the necessary devices to read different save sets concurrently. When you restore multiple dbspaces or blobspaces in parallel, you must not start the logical restore until you complete all the physical restore operations and you back up the log file that contains the record of when the restore started. Figure 7-5 on page 7-12 illustrates the concurrent restoration of two dbspaces during a warm restore.

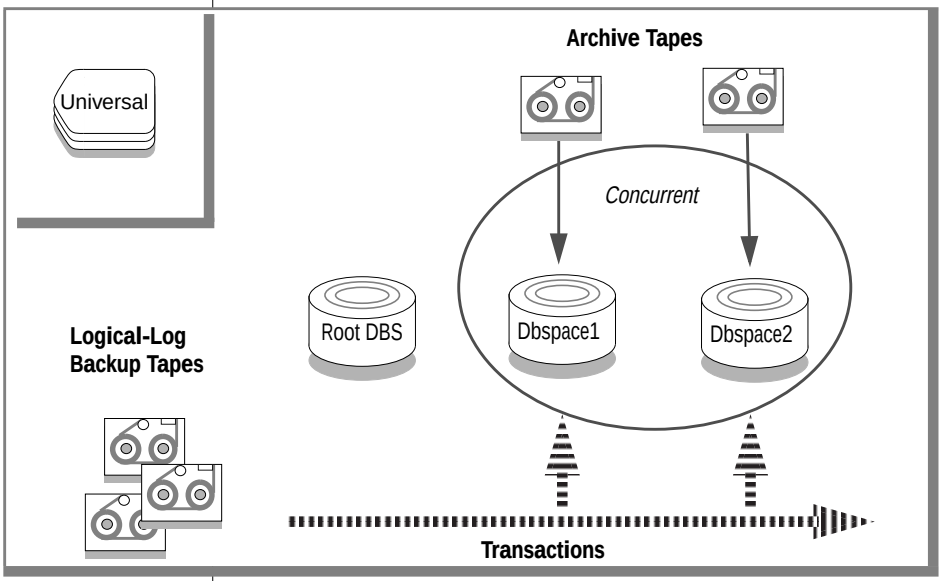


Figure 7-5
Parallelism in a Warm Restore

When you perform a cold restore, you can restore the critical dbspaces first and then restore other dbspaces in parallel. Figure 7-6 on page 7-13 illustrates this type of restore.

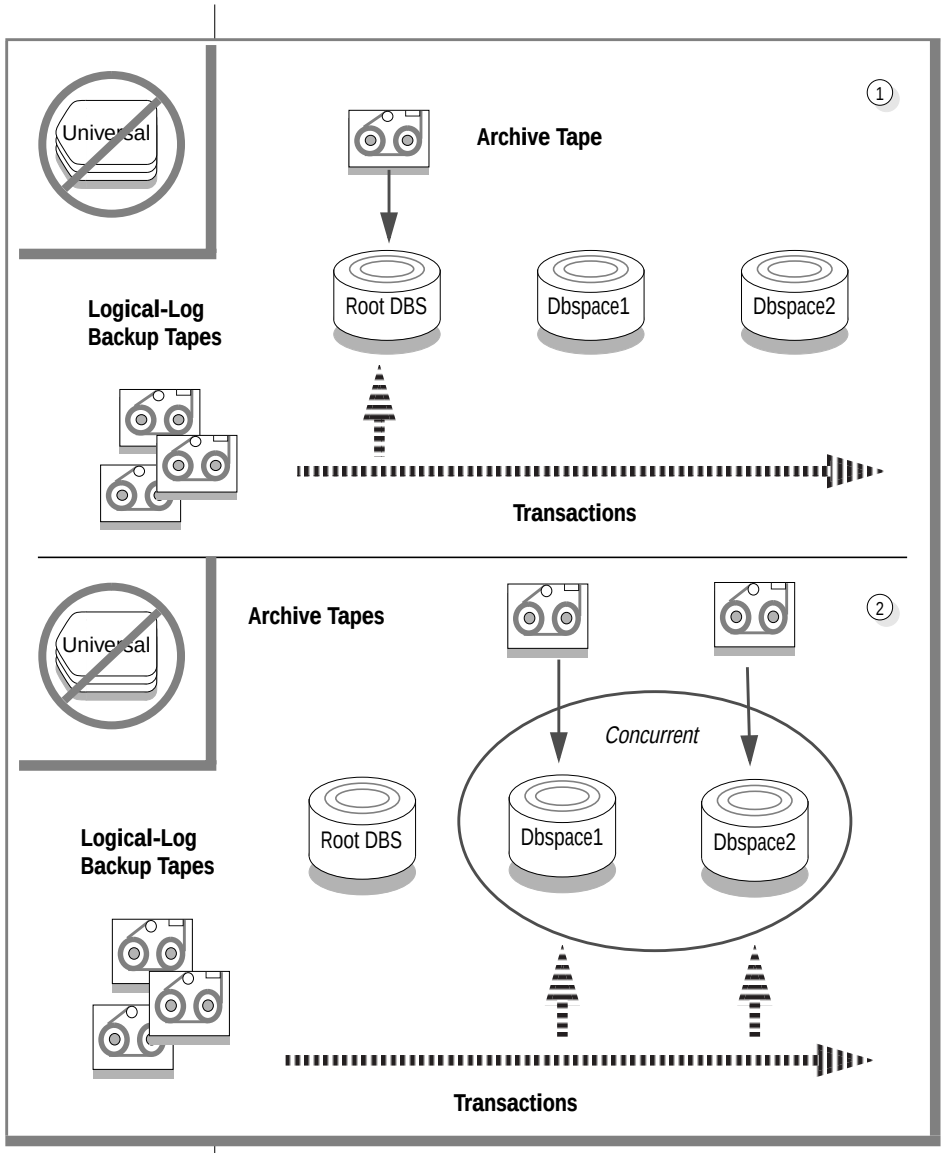


Figure 7-6
Parallelism in a Cold Restore—Critical dbspaces First, Then Other dbspaces in Parallel

Performing a Restore

This section explains how to use the RETRIEVE / DBSPACESET command to perform a physical restore and the RETRIEVE / LOGFILES command to perform a logical restore. It covers the following topics:

- Restoring all the dbspaces and blobspaces
- Restoring selected dbspaces and blobspaces

Be sure you read and understand the previous section, "Before You Restore" on page 7-3, before you attempt to restore your Universal Server data.

The RETRIEVE/DBSPACESET Command

The RETRIEVE / DBSPACESET command creates a request in the ON-Archive catalog. When executed, the request starts a physical restore for one or more dbspace sets (or individual dbspaces within a dbspace set). ON-Archive starts a physical restore from the most-recent level-0 archive and automatically prompts you to mount the volumes containing that data. After ON-Archive reads the level-0 save set, ON-Archive prompts you to mount subsequent tapes of level-1 and level-2 archive tapes of the dbspaces being restored.

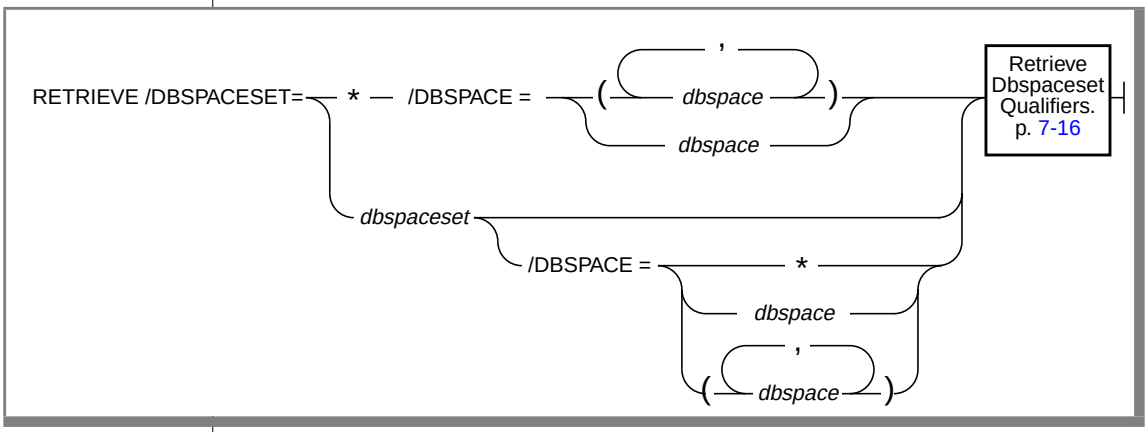
To specify a level-0 archive other than the most-recently created level-0 archive, you can use the BEFORE, SINCE, or REQUEST qualifiers.

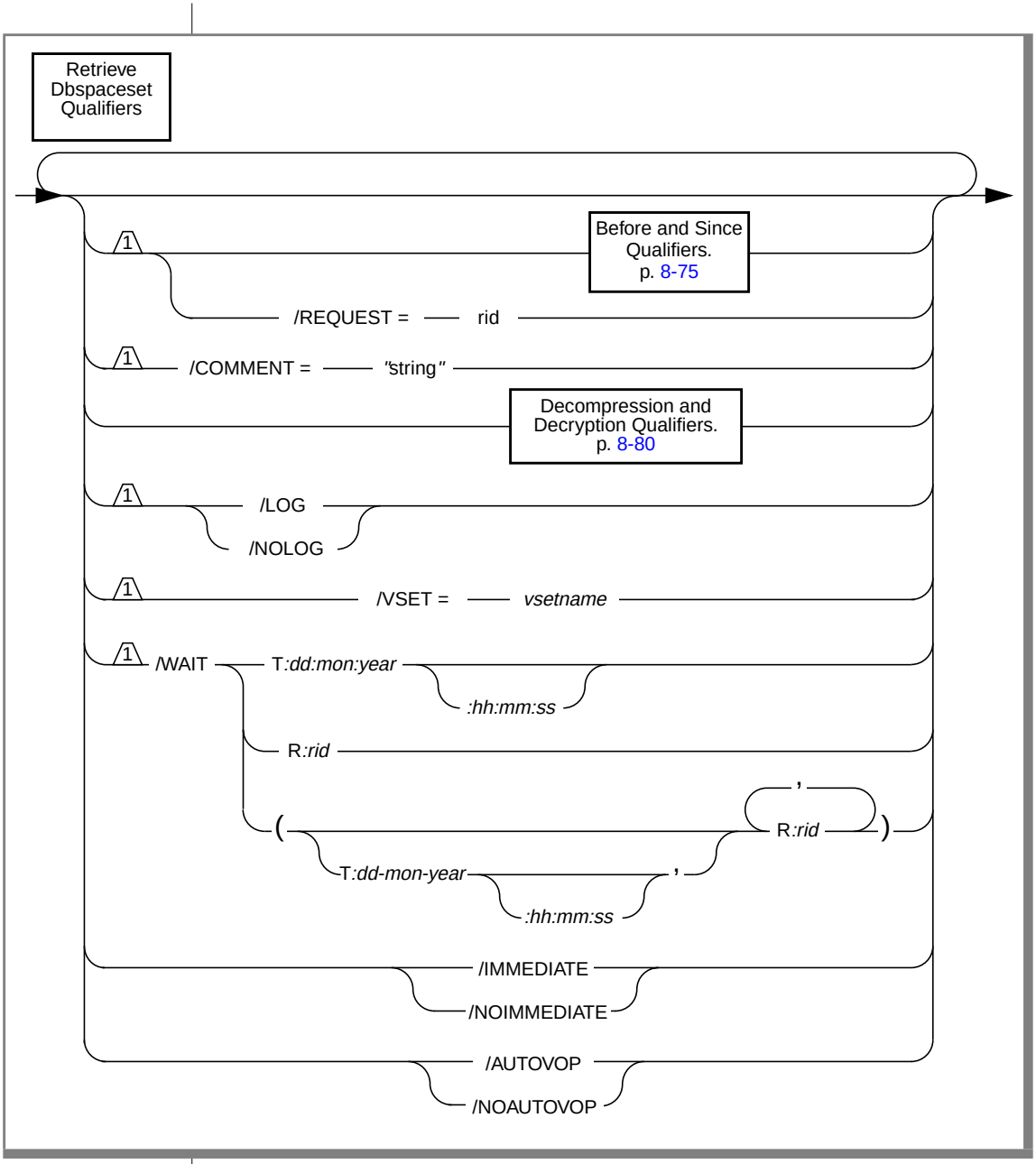
Like all commands, for the RETRIEVE / DBSPACESET command to work, Universal Server must operate in on-line or quiescent mode, **oncatlgr** must be running, and the ON-Archive catalog must be available.When these conditions do not exist—because Universal Server has experienced a catastrophic failure, or the root dbspace (containing the ON-Archive catalog) is not available—you must use **ondatartr** to perform a cold restore. See "Steps to Restore the Whole System" on page 7-22.

See "Perform a Logical Restore" on page 7-37 for a description and examples of how to use the RETRIEVE / LOGFILE command.

The following restrictions apply to the RETRIEVE / DBSPACESET command:

- Users must have read access to the vset and volumes from where they retrieve save sets.
- When ON-Archive operates in OPERATOR mode, only **informix** and **root** can retrieve dbspace sets. They can retrieve dbspace sets from archives created by any user.
- When ON-Archive operates in GROUP mode, users in the **super_archive** group and users **informix** and **root** can retrieve dbspace sets. Users in the **super_archive** group can retrieve dbspace sets from archives created by requests issued by other members of the **super_archive** group at the time the archive was created. Users **informix** and **root** can retrieve dbspace sets from archives created by any user.




Retrieve Dbspaceset Qualifiers
Before and Since Qualifiers. p. 8-75
/REQUEST =
rid
/COMMENT =
"string"
Decompression and Decryption Qualifiers. p. 8-80
/LOG
/NOLOG
/VSET =
vsetname
/WAIT
T:dd:mon:year
:hh:mm:ss
R:rid
T:dd-mon-year
:hh:mm:ss
/IMMEDIATE
/NOIMMEDIATE
/AUTOVOP
/NOAUTOVOP

AUTOVOP	specifies when you want to proceed to other tasks without waiting for a request to finish.
COMMENT	specifies a comment about the retrieve request. *string* is a text string containing a comment about the request. The string cannot exceed 80 characters in length.
DBSPACE	specifies a particular dbspace or blobspace within a dbspace set for ON-Archive to retrieve. *dbspace* is the name of a dbspace or blobspace within the dbspace set to restore. When you do not use * for DBSPACESET, the default for DBSPACE is *, and that restores all the dbspaces in the specified dbspace set.
DBSPACESET	specifies the dbspace set to restore. *dbspaceset* is the name of a dbspace set. When you use *, you must have archived all the data managed by your Universal Server database server using the ARCHIVE / DBSPACESET=* command. Also, when you use *, you cannot restore the root dbspace. You can restore the root dbspace only as part of a cold restore.
IMMEDIATE	simplifies the process of creating and executing a request. The IMMEDIATE qualifier saves time by allowing you to create and execute a request using a single command. Displays the request ID associated with that request entry.
LOG	specifies whether you must create a log file when you execute this request.
NOAUTOVOP	specifies the default setting for the RETRIEVE / DBSPACESET command. Allows you to override an AUTOVOP default setting, specified in a personal default file, to ensure that the RETRIEVE / DBSPACESET command functions as in earlier versions.

NOIMMEDIATE	specifies the default setting for the RETRIEVE / DBSPACESET command. Allows you to override an IMMEDIATE default setting, specified in a personal default file, to ensure that the RETRIEVE / DBSPACESET command functions as in earlier versions.
NOLOG	specifies that you must not create a log file when you execute the request.
REQUEST	specifies the request id of an ARCHIVE request to be restored. *rid* must exist for the creation of a level-0 archive. When you omit this qualifier, the RETRIEVE / DBSPACESET command restores the most-recent level-0 archive.
VSET	specifies the vset to use in the physical restore. This qualifier is useful when a dbspace set has been archived on different vsets. *vsetname* specifies a valid vset name.
WAIT	specifies that you must not execute the command until the time specified has past or the request specified finishes executing. See "Wait and Repeat Qualifiers" on page 8-82.

RETRIEVE/DBSPACESET Example

When you execute the following example, it retrieves all the dbspaces in the set called **slow_restore**:

```
Onarchive> RETRIEVE/DBSPACESET=slow_restore
```

The RETRIEVE/LOGFILE Command

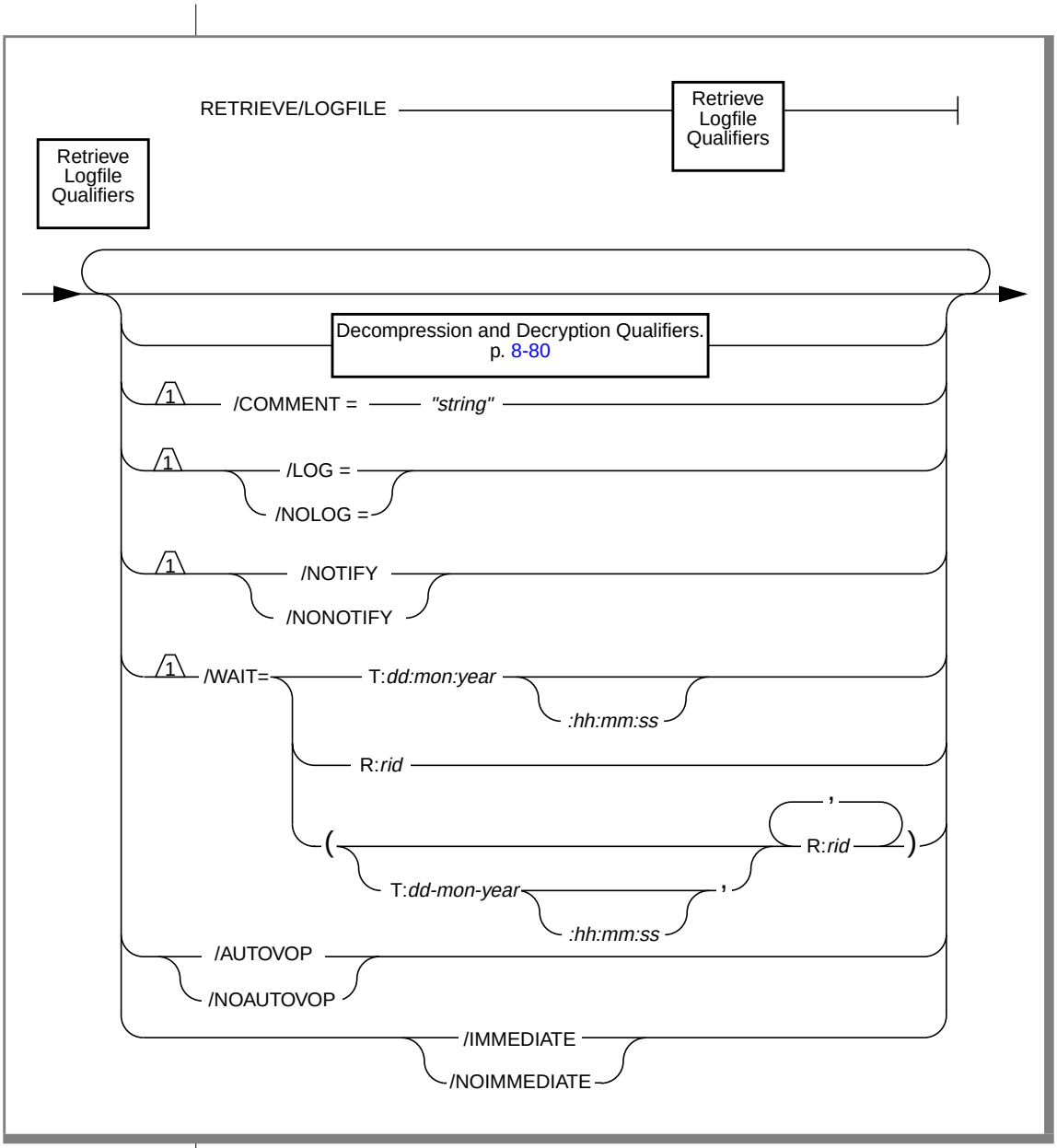
The RETRIEVE/LOGFILE command creates a request in the ON-Archive catalog, which, when executed, starts a warm logical-restore operation. As described in "Restoring Selected Dbspaces and Blobspaces" on page 7-6, the logical restore applies the logical-log records stored in the logical-log backup volumes to the data managed by Universal Server. The warm logical-restore operation only applies records to those dbspaces that were just physically restored. You must apply all the backed-up logical-log files. Universal Server has a record of the files it needs and does not make the dbspaces that were just physically restored available until it processes all the log files.

Only one logical-restore operation can occur at any time, and a logical-restore operation cannot occur when a physical-restore operation is in progress. Thus, when you execute several RETRIEVE/DBSPACESET requests, the RETRIEVE/LOGFILE command cannot execute until you finish all the physical restore operations started by the RETRIEVE/DBSPACESET commands.

See "Steps to Restore Selected Dbspaces" on page 7-33 for a description and example of how to use the RETRIEVE/LOGFILE command.

The following restrictions apply to the RETRIEVE/LOGFILE command:

- Users must have read access to the vset and volumes from which they retrieve save sets.
- When ON-Archive operates in OPERATOR mode, only **informix** and **root** can retrieve log files. They can retrieve log files from backups created by any user.
- When ON-Archive operates in GROUP mode, users in the **super_archive** group and users **informix** and **root** can retrieve log files. Users in the **super_archive** group can retrieve log files from backups created by requests issued by other members of the **super_archive** group at the time the backup was created. Users **informix** and **root** can retrieve log files from backups created by any user.


RETRIEVE/LOGFILE
Retrieve Logfile Qualifiers
Retrieve Logfile Qualifiers
Decompression and Decryption Qualifiers. p. 8-80
/COMMENT =
"string"
/LOG =
/NOLOG =
/NOTIFY
/NONOTIFY
/WAIT=
T:dd:mon:year
:hh:mm:ss
R:rid
T:dd-mon-year
:hh:mm:ss
/AUTOVOP
/NOAUTOVOP
/IMMEDIATE
/NOIMMEDIATE

AUTOVOP	specifies when you want to proceed to other tasks without waiting for a request to finish.
COMMENT	specifies a comment for this request. *string* is a text string containing a comment about the request. The string cannot exceed 80 characters in length.
IMMEDIATE	simplifies the process of creating and executing a request. The IMMEDIATE qualifier saves time by allowing you to create and execute a request using a single command. Displays the request ID associated with that request entry.
LOG	specifies that you create a log file when you execute the request. When you cannot execute the command immediately because the WAIT qualifier was used, file specifications are written to a log file in the directory where **onarchive** or **onautovop** was started. The log file is named **ARC***rid***.LOG**, where *rid* is the request id of the RETRIEVE/LOGFILE command.
NOAUTOVOP	specifies the default setting for the RETRIEVE/LOGFILE command. Allows you to override an AUTOVOP default setting, specified in a personal default file, to ensure that the RETRIEVE/LOGFILE command functions as in earlier versions.
NOIMMEDIATE	specifies the default setting for the RETRIEVE/LOGFILE command. Allows you to override an IMMEDIATE default setting, specified in a personal default file, to ensure that the RETRIEVE/LOGFILE command functions as in earlier versions.
NOLOG	specifies that you create a log file when you execute the request. It is the default.

NOTIFY	specifies that after the request executes, electronic mail is sent to the user who created the request.
NONOTIFY	specifies that the RETRIEVE/LOGFILE command sends no mail after the request executes. It is the default.
WAIT	controls whether the request must wait for certain events to occur before it executes. See "Wait and Repeat Qualifiers" on page 8-82.

RETRIEVE/LOGFILE Example

When you execute the following example, it retrieves all the relevant logical-log files after a RETRIEVE/DBSPACESET command (that is, a restore procedure) with request id 156 finishes:

```
Onarchive> RETRIEVE/LOGFILE/WAIT=R:156
```

Steps to Restore the Whole System

This section outlines the steps you need to perform during a full-system restore with ON-Archive. The following list shows the main steps in a full-system restore:

- Gather the appropriate volumes.
- Decide whether to do a complete cold or a mixed restore.
- Verify your Universal Server configuration.
- Prepare to salvage log files.
- Perform a cold physical restore.
- Start parallel cold physical restores.
- Perform a logical restore.
- Perform a restore of the remaining dbspaces and blobspaces, when you require a mixed restore.

You must read these instructions and familiarize yourself with them before you attempt a full-system restore.

Gather the Appropriate Volumes

To restore your Universal Server database server, you must gather the appropriate archive and logical-log volumes containing the Universal Server data. When multiple save sets reside on your volumes, you also must know the identification number of the save sets you are going to restore. When you do not know the numbers from the volume labels, you can find them out before you restore. You can use **ondatartr** to list the contents of a tape.

After performing archives and backups, you can use the LIST/DBSPACESET and LIST/LOGFILE commands with the OUTFILE qualifier to create text files that list the vsets and volumes of the archives and log files. This ensures that the information is available independent of the archive catalog.

Archive Volumes

Before you start your restore, gather together all the volumes (tapes) from your latest level-0 archive containing the dbspaces and blobspaces you are restoring, and the latest subsequent level-1 or level-2 archives. When you make level-0 archives of different dbspaces and blobspaces created at different times, you can use them, provided that the level-0 archive containing the root dbspace was made before all the other level-0 archives you use.

Identify the volume that has the level-0 archive of the root dbspace on it; you must use this volume first.

Logical-Log Volumes

Gather together all the logical-log volumes after the oldest level-0 archive of the dbspaces and blobspaces you are restoring.

When you have dbspaces and blobspaces that have different times for their latest level-0 archive, you must use all the logical-log volumes after the oldest of those archives.

Decide on a Cold or a Mixed Restore

As mentioned in "Choosing a Universal Server Mode—Cold, Warm, or Mixed Restore" on page 7-6, when you are restoring your entire Universal Server database server, you can restore the critical dbspaces (and any other dbspaces or blobspaces you want to come on-line quickly) during a cold restore and then restore the remaining dbspaces and blobspaces during a warm restore. You must decide before you start the restore whether you want it entirely cold, or mixed. (See "Choosing a Universal Server Mode—Cold, Warm, or Mixed Restore" on page 7-6.)

Verify Your Universal Server Configuration

During a cold restore, you cannot reinitialize shared memory, add chunks, or change tape devices. Thus, when you begin the restore, the current Universal Server configuration must be compatible with, and accommodate, all ONCONFIG parameter values assigned after the most-recent archive.

For guidance, use the copies of the configuration file that you create at the time of each archive. However, do not blindly set all current parameters to the same values as were recorded at the last archive. Pay attention to the following groups of parameters:

- Shared-memory parameters
- Mirroring parameters
- Device parameters

Set Shared-Memory Parameters to Maximum Assigned Value

Verify that you set your current shared-memory parameters to the *maximum* value assigned after the level-0 archive. For example, when you decrease the value of USERTHREADS from 45 to 30 sometime after the level-0 archive, you must begin the restore with USERTHREADS set at 45, and not at 30, even though the ONCONFIG file for the last archive could set the value of USERTHREADS at 30. (When you do not have a record of the maximum value of USERTHREADS after the level-0 archive, set the value as high as you think necessary. You could reassign values to BUFFERS, LOCKS, and TBLSPACES as well because the minimum values for these three parameters are based on the value of USERTHREADS.)

Set Mirroring Configuration to Level-0 Archive State

Verify that your current mirroring configuration matches the configuration that was in effect at the time of the level-0 archive you are using of the root dbspace. Because Informix recommends that you create a level-0 archive after each change in your mirroring configuration, this must not be a problem. The most critical parameters are the mirroring parameters that appear in the Universal Server configuration file, MIRRORPATH and MIRROROFFSET.

Ensure That Needed Devices Are Available

Verify the availability of raw devices or files used for Universal Server storage (of the dbspaces and blobspaces being restored) after the level-0 archive.

For example, when you drop a dbspace or mirroring for a dbspace after your level-0 archive, you must ensure that the dbspace or mirror chunk device is available to Universal Server when you begin the restore. When Universal Server attempts to write to the chunk and cannot find the chunk, the restore does not complete. Similarly, when you added a chunk after your last archive, you must ensure that the chunk device is available to Universal Server when it begins to roll forward the logical log.

Prepare to Salvage Log Files

Informix recommends that you salvage your logical-log files on disk when performing a cold restore. Salvaging logs allows you to recover the last transactions in the logical log that were not backed up. See the SALVAGELOGS qualifier for the **ondatartr** version of the "RETRIEVE / DBSPACESET Command" on page 9-24.

When you do a cold restore without salvaging the logs, you could also lose the following information:

- Information about both archive and backup requests (because the transactions against the archive catalog are lost when you do not reprocess the logs)
- The archive checkpoint

For example, imagine that you back up logs 1 to 5, archive a dbspace set, and then experience a system failure. When you perform a cold restore without salvaging the log files, you can lose both the checkpoint for the last archive and the archive catalog transactions for it because they were in log 6.

Furthermore, when you do not salvage and reprocess the logs, the transaction that marks the last logical-log backup request as COMPLETED invariably becomes lost. Thus when **oncatlgr** starts up, it notices that the log backup request is marked as EXECUTING and it marks the request as FAILED.

When you cannot salvage the logical-log files during a cold restore, always catalog the last log-backup volume when Universal Server is back on-line. *Do not*, however, catalog any missing or failed archive requests.

You specify that you must salvage log files when you perform the cold restore. You must save the salvaged logs to a device that is the same type (disk or tape) as your regular logical-log backups, and you must write them to a separate volume. Thus, you must ready a blank volume (when your log vset is a tape vset), or you must create a directory for the salvaged logs (when your log vset is a disk vset).

Important: *When you do a parallel cold restore, specify the SALVAGELOGS qualifier only for the restore of the critical dbspaces, and those are the first dbspaces to be restored in a parallel cold restore. Do not specify SALVAGELOGS for the other dbspaces that you are restoring in parallel. See the* *"RETRIEVE/DBSPACESET Command" on page 9-24 in "The ondatartr Utility"* *section.*

Perform a Cold Physical Restore

To perform a cold restore, Universal Server must operate in off-line mode.

To perform a cold physical restore, you use the **ondatartr** utility. You must log in as **informix** or **root** to use **ondatartr**. The syntax and commands for **ondatartr** are found in "The ondatartr Utility" on page 9-12. To start **ondatartr**, enter the following command at your operating-system prompt:

```
% ondatartr
ONDATARTR>
```

List Volume Contents

When you do not know the save-set identification numbers of the save sets you want to restore, you can mount the volumes on a device and list the save sets on them using the **ondatartr** commands LIST/DISK or LIST/TAPE command, as shown in the following example:

```
ONDATARTR> LIST/TAPE=(/dev/rst0)
```

See the "LIST/DISK Command" on page 9-20 or "LIST/TAPE Command" on page 9-22 for more information on how to list volume contents.

Retrieve Dbspaces

You retrieve the dbspaces (or blobspaces) from the tape using the **ondatartr** RETRIEVE/DBSPACESET command, as shown in the following example. (See "The RETRIEVE/DBSPACESET Command" on page 7-14.)

```
ONDATARTR> RETRIEVE/DBSPACESET=*/TAPE=(/dev/rst0)/SALVAGELOGS=(/dev/rst1)
```

You can only retrieve DBSPACESET=* if you archived all the dbspaces and blobspaces managed by your Universal Server database server using DBSPACESET=* in your archive request.

When you do not use DBSPACESET=*, you must include at least the root dbspace in this initial cold physical-restore operation. Before you can start any secondary **ondatartr** processes, you must restore the reserved pages and Universal Server must run in fast-recovery mode. See “Start Parallel Cold Physical Restores”.

As shown in Figure 7-7 on page 7-29, the first thing the restore procedure does is salvage logical-log files.

After you salvage the logs, and you are using a tape volume, mount the volume containing the latest level-0 archive of the root dbspace on a tape drive.

After you restore the level-0 archive, **ondatartr** asks whether you made archives of other levels to restore.

Start Parallel Cold Physical Restores

When you do not restore using DBSPACESET=*, you can start other **ondatartr** processes to restore dbspaces archived in other save sets after the following message appears in the Universal Server message log:

```
Recovery Mode
```

Figure 7-7 is an example of a cold restore using **ondatartr**:

Figure 7-7
Example of a Cold Restore Using ondatartr

```
ONDATARTR>RETRIEVE/DBSPACESET=*/TAPE=(/dev/rst1)/SALVAGELOGS=(/dev/rst0)

Executing...

Log files 111 through 115 are eligible for salvaging.
What log file number should ONDATARTR start with? : 114

Please mount volume 1 of ONDATARTRLOG and press < RETURN > ** WRITE **

Save set 1345 being written to vset ONDATARTRLOG.

Logfile 114 backed up.

Logfile 115 backed up.

Mount the volume with the archived data.
Press the return key when ready.
What saveset ID is to be used on volume /dev/rst1?: 45
Level 0 physical restore started.

The file rootdbs has been retrieved.

The file dbs1 has been retrieved.

The file dbs2 has been retrieved.

End of level 0 archive reached on this volume.
Does this saveset continue on another volume? (Y/N): n
Do you have a level 1 archive to retrieve? (Y/N): y
Mount the volume with the archived data.
Press the return key when ready.
What saveset ID is to be used on volume /dev/rst1?: 48
Level 1 physical restore started.

The file rootdbs has been retrieved.

The file dbs1 has been retrieved.

The file dbs2 has been retrieved.

End of level 1 archive reached on this volume.

Does this saveset continue on another volume? (Y/N): n

Do you have a level 2 archive to retrieve? (Y/N): y
Mount the volume with the archived data.
Press the return key when ready.
What saveset ID is to be used on volume /dev/rst1?: 49
Level 2 physical restore started.

The file rootdbs has been retrieved.

The file dbs1 has been retrieved.

The file dbs2 has been retrieved.
```

Perform a Logical Restore with ondatartr

After you finish with all cold physical-restore operations, Universal Server is in fast-recovery mode.

When you perform a mixed restore, or when you perform a complete cold restore from more than one archive, you must perform a cold logical restore at this point to ensure that all dbspaces are restored to the same point in time. When you restore Universal Server dbspaces from more than one archive, either full-system or selected-dbspace archives, each archive contains a set of dbspaces that was saved at a different point in time. Following the restore, the only way to guarantee that each dbspace possessed the same set of transactions applied to it is to apply the logical log to all dbspaces. To apply the logical log to all dbspaces, you need all logical-log backups after the earliest archive from which you are restoring.

Important: *You can perform point-in-time recovery by using the* ***ondatartr*** *utility. You use the UNTIL qualifier of the RETRIEVE/LOGFILE command as specified in* *"RETRIEVE/LOGFILE Command" on page 9-27*.

Warning: *When you restore from multiple archives, and you are missing any logical-log files between the earliest archive and the latest archive, the only way to ensure that all dbspaces are restored to the same point in time is to restore from an earlier complete set of archives and log files. For example, when the earliest archives you want to restore were created when the active log file was log 10, you need logical-log file 10 for the logical restore. When you are unable to locate the backup of log file 10 at the time of the restore, the only solution is to restore from an earlier, complete set of archives and matching logical-log files.*

When you restore all the dbspaces and blobspaces from a single full-system archive, you can either perform a cold logical restore or, when you do not have logical-log files to restore, you can bring Universal Server into on-line mode (by executing **onmode -m**). When you do not do a logical restore, however, your data remains in the state it was in at the time of the last archive.

When you archive all Universal Server dbspaces in a level-0, whole-system archive, Universal Server stores a snapshot of the logs with the archive. When you restore from a single, full-system archive, any unresolved transactions *at the time of the archive* are resolved during the restore using the log-file snapshot. Note that you take the snapshot only for a level-0, full-system archive and that the restore only restores Universal Server to the state it was in at the time of the archive. The logical logs are required to restore any transactions after the time of the archive. (For the effect of setting LTAPEDEV to **/dev/null**, see the Warning in "When You Do Not Need to Recover" on page 6-5.)

List Volume Contents

When you start the logical restore, **ondatartr** knows the specific log file it needs to start with. It tells you the log-file number, and you must supply it with the save-set number where the log file is located. You must put this information on your tape labels, but you can also use the LIST command to obtain it.

Retrieve Log Files and Salvaged Logs

You use the **ondatartr** RETRIEVE/LOGFILE command to perform the logical restore. (See "RETRIEVE/LOGFILE Command" on page 9-27.) When **ondatartr** comes to the end of the regular logical-log files, it give you the opportunity to retrieve the salvaged logical-log files. During a cold physical restore with SALVAGELOGS set, **ondatartr** displays its process id as the save-set id of the salvaged logical-log files. At the end of the logical restore, **ondatartr** prompts you to enter this save-set id to retrieve the salvaged logical-log files. Refer to Figure 7-7 on page 7-29 to see where **ondatartr** displays the save-set id for the salvaged logical-log files.

In Figure 7-8, the log files number 10 and 11 are in save set 22, log files 12 and 13 are in save set 26, and log files 14 and 15 were salvaged by an **ondatartr** process with a pid of 1345 (which is the save-set number).

Figure 7-8
Example of a Logical Restore

```
ONDATARTR>RETRIEVE/LOGFILE/TAPE=(/dev/rst1)

Executing...
Logical restore started with log number: 10.

Mount the volume with the archived data.
Press the return key when ready.
What saveset ID is to be used on volume /dev/rst1?: 22
The file LF00000010 has been retrieved.

The file LF00000011 has been retrieved.

End of saveset reached on this volume.
Do you have more log backups to process? (Y/N): y
Mount the volume with the archived data.
Press the return key when ready.
What saveset ID is to be used on volume /dev/rst1?: 26
The file LF00000012 has been retrieved.

The file LF00000013 has been retrieved.

Do you have more log backups to process? (Y/N): y
Mount the volume with the archived data.
Press the return key when ready.
What saveset ID is to be used on volume /dev/rst1?: 1345
The file LF00000014 has been retrieved.

The file LF00000015 has been retrieved.

Does this saveset continue on another volume? (Y/N): n
Do you have any more log backups to process (Y/N): n
```

At the end of the logical restore, Universal Server operates in quiescent mode. You can bring Universal Server into on-line mode at this point (by executing **onmode -m**) and continue processing as usual.

When you perform a mixed restore, you can start a warm restore of the remaining dbspaces and blobspaces after you bring Universal Server into on-line mode.

Steps to Restore Selected Dbspaces

This section outlines the steps you need to perform to restore selected dbspaces and blobspaces with ON-Archive. Perform these steps while Universal Server is in on-line or quiescent mode (a warm restore). The following list shows the main steps in a warm restore:

- Gather the appropriate volumes.
- Verify your Universal Server configuration.
- Catalog and copy the salvaged logs when you perform a mixed restore.
- Perform a warm physical restore.
- Perform multiple warm physical restores.
- Back up the logical-log files.
- Perform a warm logical restore.

Read these instructions to familiarize yourself with them before you attempt a warm restore. Also see "When You Do Not Need to Recover" on page 6-5 for the implications of not backing up your logical-log files.

To perform a warm restore, Universal Server must operate in on-line mode.

Gather the Appropriate Volumes

You must gather the appropriate archive and logical-log volumes before you begin to restore the selected dbspaces and blobspaces. When you want to know the specific dbspaces you must restore to restore a particular database, the LIST/DATABASE command shows you the correlation of the tables, dbspaces, and dbspace sets for a given database.

ON-Archive tells you the vsets and volumes you must mount during a warm restore. In the case of disk vsets, ON-Archive automatically retrieves the save sets without requiring any information from the operator.

You can use the LIST/DBSPACESET and LIST/LOGFILE commands to see those volumes and vsets that hold archives and log files.

A warm restore can execute unattended when you mount the correct volumes before you start. Informix recommends that you use the WAIT=T: qualifier when you do a warm restore to ensure that the backup does not execute until all warm physical restores complete, and that the logical restore does not execute until the backup completes.

Archive Volumes

Before you start your restore, gather together all the volumes (tapes) from your latest level-0 archive containing the dbspaces and blobspaces you are restoring, and the latest subsequent level-1 or level-2 archives. When level-0 archives of different dbspaces and blobspaces were created at different times, you can use them.

When you restore selected dbspaces and blobspaces as part of a mixed restore, you cannot use level-0 archives older than the level-0 archive containing the root dbspace used in the cold restore.

Logical-Log Volumes

Gather together all of the logical-log volumes after the oldest level-0 archive of the dbspaces and blobspaces you are restoring.

When you create dbspaces and blobspaces at different times for their latest level-0 archive, you must use all the logical-log volumes after the oldest of those archives.

When you perform a mixed restore, you must use all the logical-log volumes created after the level-0 archive of the root dbspace used in the cold restore. This statement is true even when you restore dbspaces or blobspaces that have a level-0 archive more recent than the level-0 archive of the root dbspace.

Verify Your Universal Server Configuration

During a warm restore, you do not need to worry about shared-memory parameters, as you do for cold restores.

Ensure That Needed Devices Are Available

Verify the availability of the raw devices or files that were used for Universal Server storage (of the dbspaces and blobspaces being restored) after the level-0 archive.

For example, when you drop a dbspace or mirroring for a dbspace after your level-0 archive, you must make the dbspace or mirror chunk device available to Universal Server before you begin the restore. When Universal Server attempts to write to the chunk and cannot find the chunk, the restore does not complete. Similarly, when you add a chunk after your last archive, you make the chunk device available to Universal Server before it begins to roll forward the logical-log files.

Catalog and Copy Salvaged Logs

When you perform a mixed restore and salvage your logical-log files during the cold restore, after Universal Server is on-line you must catalog the save set created during the salvage operation. Then you must copy the save set to the vset that contains the rest of your logical-log files before you perform the warm restore. When you do not catalog and copy the salvaged logs, ON-Archive cannot use them during the logical restore because it can only perform one logical restore and can only use save sets from one vset for the operation.

The procedure is the same for handling the log files created in an emergency backup, as explained in "Performing Emergency Logical-Log File Backups" on page 6-13.

Perform a Warm Physical Restore

To perform a warm physical restore, you use the RETRIEVE/DBSPACESET command with **onarchive**. See the section "The RETRIEVE/DBSPACESET Command" on page 7-14 for the syntax of the command.

The following example retrieves a dbspace called **disk2** from a save set that includes all the dbspaces managed by Universal Server:

```
Onarchive> RETRIEVE/DBSPACESET=*/DBSPACE=disk2
```

ON-Archive prompts you to mount the appropriate volumes on devices, all the way through the physical restore.

Perform Multiple Warm Physical Restores

When you archive dbspaces in different save sets, you can perform warm physical restores in parallel by executing requests to restore different dbspaces at the same time.

Back Up the Logical-Log Files

Before you start the logical restore (which must occur for the dbspaces and blobspaces being restored to become available), you must back up the logical-log file that was current when the warm physical restore started. This log file is usually the same as the current log file, but perhaps not. For example, when log file 7 was current when the warm restore began, but transactions were posted during the warm restore and filled up the log file so that log file 8 is now the current log file, you must back up log file 7. See "Backing Up All Full Logical-Log Files" on page 6-11 or "Backing Up the Current Log File" on page 6-11.

Perform a Logical Restore

After you complete all the warm physical-restore operations and you back up the logical-log files, you must perform a logical restore. (Unlike a cold restore, where you can choose not to restore logical-log files, you must restore the log files during a warm restore.)

To perform a logical restore after a warm physical restore, you must reprocess the following logical-log files:

1. The log that contains the checkpoint for the archive that was restored during the warm restore
2. The log that contains the checkpoint for the warm physical restore
3. All the logs between item 1 and item 2

When Universal Server cannot locate any of these logs, the logical restore fails.

To perform a logical restore, you use the RETRIEVE/LOGFILE command with **onarchive**. See the section "The RETRIEVE/LOGFILE Command" on page 7-19 for the syntax of the command.

```
Onarchive> RETRIEVE/LOGFILE
```

ON-Archive prompts you to mount the appropriate volumes.

When the logical restore finishes, the dbspaces and blobspaces being restored become available.

Chapter

8

Administering ON-Archive

This chapter describes the remainder of the ON-Archive commands and qualifiers. In general, the commands in this chapter enable you to display, catalog, modify, or remove information in the ON-Archive catalog.

The following table tells you where to find commands that this chapter does *not* describe:

Command	Purpose	Page
ARCHIVE	Creates an archive of Universal Server data.	5-12
BACKUP	Backs up logical-log files.	6-9
EXECUTE	Executes a specific request, or all requests for a vset.	4-27
EXIT	Terminates ON-Archive.	4-17
HELP	Gets help on an ON-Archive topic.	4-23
MENU	Uses the menu interface.	4-18
RETRIEVE / DBSPACESET	Restores the specified dbspaces from an archive.	7-14
RETRIEVE / LOGFILE	Starts a warm logical restore operation.	7-19

Working with Requests

ON-Archive provides the following commands for administering requests that were entered previously and reside in the catalog:

Command	Purpose
CANCEL	Cancels a request in the catalog.
LIST / REQUEST	Displays information on all requests currently in the catalog.
MODIFY / COMMAND	Modifies a request in the catalog.
REMOVE / FAILED_REQUEST	Removes requests with a status of FAILED from the catalog.
REMOVE / REQUEST	Removes requests with a status of CANCELLED or COMPLETED from the catalog.

This section describes the syntax and use of each of these commands. See "Creating and Executing Requests" on page 4-26 for a list of the ON-Archive commands that become requests and for information on how to create and execute them.

The CANCEL Command

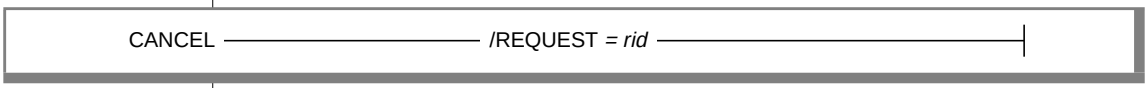
The CANCEL command changes the status of a request from NEW or EXECUTE to CANCELLED. Changing request status to CANCELLED stops the execution of a request or prevents the request from being executed.

You can cancel a request only when its status is NEW or EXECUTING. When the request status is CANCELLED, FAILED, UNCOMPLETED, or COMPLETED, the CANCEL command has no effect. (To remove requests from the catalog, see "The REMOVE / FAILED_REQUEST Command" on page 8-15 and "The REMOVE / REQUEST Command" on page 8-17.)

If you execute the request when you issue the CANCEL command, ON-Archive stops its execution as soon as the request has been cancelled (that is, before any major I / O operations). Any work done before you execute the CANCEL command remains.

The following restrictions apply to the CANCEL command:

- When ON-Archive is running in OPERATOR privilege mode, only **root** and **informix** can cancel requests. They can cancel requests made by any user.
- When ON-Archive is running in GROUP privilege mode, users in the **super_archive** group can cancel requests of other users in the same group.



REQUEST specifies the identification number of the request to cancel.

rid is a request id in the catalog. See "Changing Database Logging Status" on page 8-62 to find out request ids.

CANCEL Example

The following example cancels request number 357:

```
Onarchive> CANCEL/REQUEST=357
```

The LIST/RECOVERY Command

The LIST/RECOVERY command improves the data-restoration process and generates a report that displays data-restoration information. The LIST/RECOVERY command provides a simple mechanism for printing or saving. In addition, the LIST/RECOVERY command generates a report that does the following tasks:

- Sorts volumes in the mount order required for restore
- Provides an indication of the volumes that contain critical data
- Serves as a guide during cold restores to make Universal Server available as quickly and easily as possible.

The LIST/RECOVERY command sorts the data in the order required to restore the data.When you specify a dbspace, or a list of dbspaces, LIST/RECOVERY displays only the data required to perform a warm restore of the dbspaces.

When you run ON-Archive in OPERATOR mode, you can list the recovery report only when you log in as **informix** or **root**. When you run ON-Archive in GROUP mode, only users in the **super_archive**, **informix**, or **root** groups can generate a LIST/RECOVERY report.

*	is a wildcard character that represents all dbspace names.
DBSPACE	instructs the LIST command to display dbspace information.
dbspacename	is the name of a specific dbspace.
RECOVERY	instructs the LIST command to display data-restoration information.

Execute the LIST/RECOVERY command after the following events:

- Archiving the entire installation
- Archiving critical and noncritical dbspaces separately
- Archiving some critical and noncritical dbspaces together, but without archiving the entire installation
- Archiving level 0, changing data, then archiving again at levels 1 and 2
- Archiving, changing data, and backing up the logical logs
- Archiving successfully, trying another archive that fails, then archiving successfully
- Making incremental archives on specific dbspacesets

When those events finish, verify the contents of the ON-Archive activity log. In addition, make sure that you check the contents of the ON-Archive activity log after you complete the following tasks:

- Explicitly define dbspacesets, volume sets, and volumes
- Create requests to archive and back up without executing those requests
- Restore a dbspace
- Enact a cold restore

For more information on the ON-Archive activity log, see "Adding an ON-Archive Activity Log to Log Archive Events" on page 9-37.

LIST/RECOVERY Example

The following example shows one way to display recovery information using the LIST/RECOVERY command:

```
LIST/RECOVERY/DBSPACE=*
```

The output displays recovery information for all dbspaces in an installation that performs whole archives daily at approximately 1 a.m., level 0 archives monthly, level 1 archives Sunday mornings, level 2 archives daily, and continuous log backups (three logs per saveset) between archives.

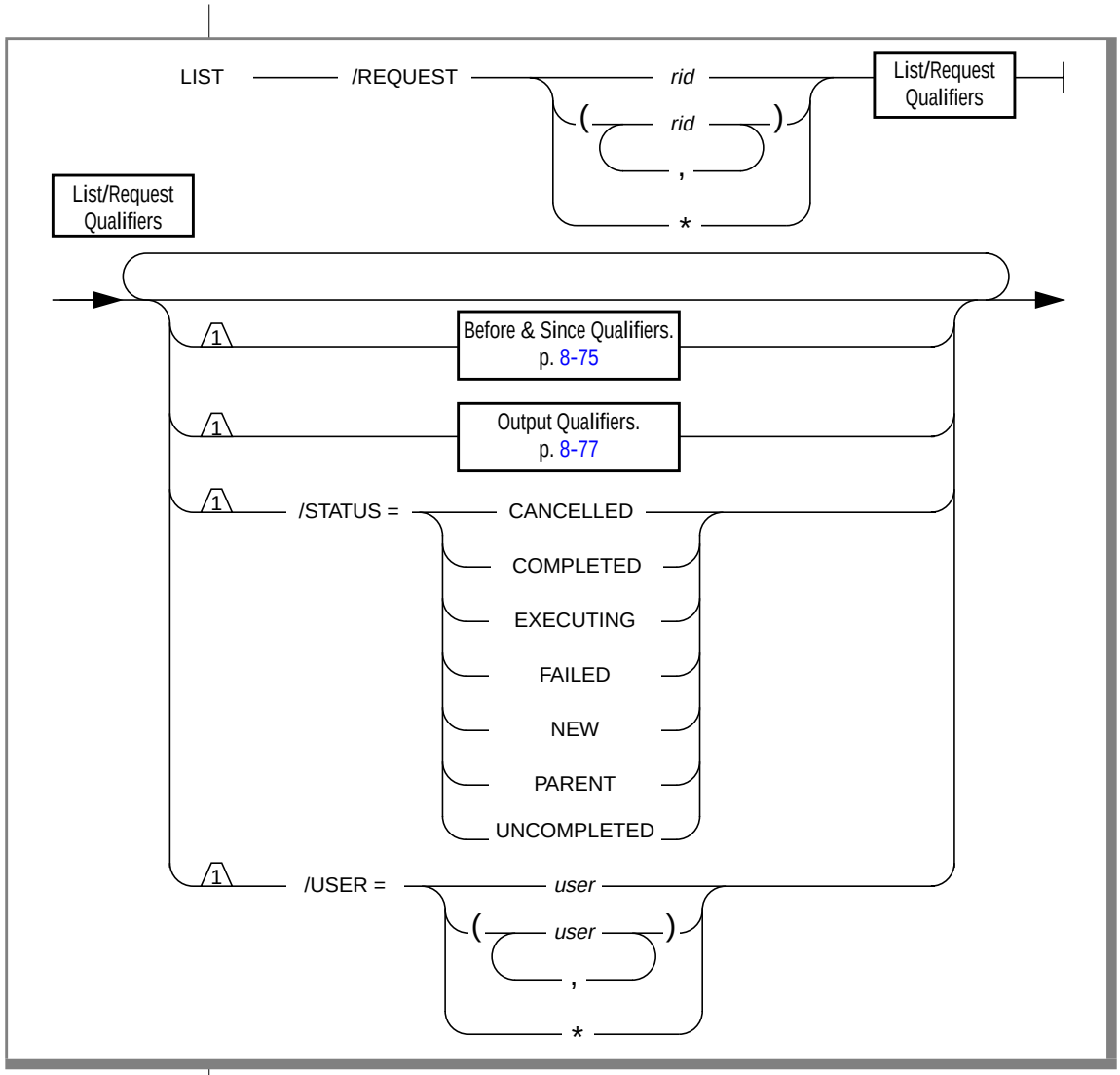
```
onarchive> LIST/RECOVERY
Recovery requirements as of April 18, 1994 at 09:20 for * + = required for minimal restore
Vol   Save Set  VSet              Label   Device   Date               Level
0100  00000100  March_Monthly     Mar0M   8mmTape  01-APR-1994 01:13  0 dbs1, dbs2, dbs3, rootdbs+
0101  00000100  March_Monthly     Mar1M   8mmTape  01-APR-1994 01:13  0 dbs1, dbs2, dbs3, rootdbs+
0102  00000100  March_Monthly     Mar2M   8mmTape  01-APR-1994 01:13  0 dbs1, dbs2, dbs3, rootdbs+
0134  00000130  April_Week_2      Apr0W2  8mmTape  17-APR-1994 01:09  1 dbs1, dbs2, dbs3, rootdbs+
0135  00000132  April_Week_2      Apr1W2  8mmTape  17-APR-1994 01:25  2 dbs1, dbs2, dbs3, rootdbs+
0137  00000132  April-Day_18_Logs Apr18L  8mmTape  18-APR-1994 05:47  Backup LF00001234, LF00001235, LF00001236
0140  00000134  April-Day_18_Logs Apr18L  8mmTape  18-APR-1994 09:15  Backup LF00001237, LF00001238, LF00001239
```

The LIST/REQUEST Command

The LIST / REQUEST command displays information on all past and current requests in the catalog. You can create a request with the ARCHIVE, BACKUP COPY, or RETRIEVE command.

The following restrictions apply to the LIST / REQUEST command:

- When ON-Archive is running in OPERATOR mode, only **root** or **informix** can display information about requests. They can display information about all requests.
- When ON-Archive is running in GROUP mode, users belonging to the **super_archive** group can display information about all requests issued by members of the group. Users **informix** and **root** can list requests of all users.


LIST
/REQUEST
rid
rid
,
*
List/Request Qualifiers
List/Request Qualifiers
1
Before & Since Qualifiers. p. 8-75
1
Output Qualifiers. p. 8-77
1
/STATUS =
CANCELLED
COMPLETED
EXECUTING
FAILED
NEW
PARENT
UNCOMPLETED
1
/USER =
user
user
,
*

REQUEST specifies the requests designated for display.

rid is a request id in the catalog.

* specifies all requests that the user created.

STATUS displays only those requests with the specified status. See "Request Statuses" on page 4-29 for information on the request status.

USER specifies other users who created requests. Whether you can see requests created by other users is subject to the privileges you possess and the restrictions that apply to the LIST/REQUEST command.

user is a valid operating system user id.

* specifies all users.

LIST/REQUEST Examples

The following example displays a brief list of all requests submitted by the user issuing the LIST command:

```
Onarchive> LIST/REQUEST=*/BRIEF
```

```
RID         Username       Command   Issue Date    Status      Former   RID/SID
--------    ------------   --------  -----------   ---------   ----------------
00000010    ARCHIVE_TEST   ARCHIVE   12-JUL-1993   COMPLETED   *******  *******
00000011    ARCHIVE_TEST   RETRIEVE  13-JUL-1993   COMPLETED   *******  *******
00000014    ARCHIVE_TEST   COPY      20-JUL-1993   NEW         *******  *******
00000128    ARCHIVE_TEST   ARCHIVE   02-OCT-1993   COMPLETED   *******  *******
```

The following example lists all requests submitted by the user at least 120 days before yesterday:

```
Onarchive> LIST/REQUEST=*/BRIEF/BEFORE=(YESTERDAY - 120)
```

```
RID         Username       Command    Issue Date    Status      Former   RID/SID
--------    ------------   --------   -----------   ---------   ----------------
00000010    ARCHIVE_TEST    ARCHIVE    12-JUL-1993   COMPLETED   ******* *******
00000011    ARCHIVE_TEST    RETRIEVE   13-JUL-1993   COMPLETED   ******* *******
00000014    ARCHIVE_TEST    COPY       20-JUL-1993   NEW         ******* *******
```

This example lists all requests submitted by the user issuing the LIST command. It shows all the information about the requests.

```
Onarchive> LIST/REQUEST=*/FULL
```

```
Request ID      : 00000010
Username        : ARCHIVE_TEST
Directory       : /usr/archive_test
Former RID      : ***********Former SID:***
Issue Date      : 12-JUL-1993 10:57:13Expiry Date:
Exec. date      : 12-JUL-1993 11:38:56
Status          : COMPLETED
Command         : ARCHIVE/DBSPACESET=*
```

The MODIFY/COMMAND Command

The MODIFY/COMMAND command enables you to modify previously entered ARCHIVE, BACKUP, COPY, REMOVE (with WAIT), and RETRIEVE requests that display a status of NEW, FAILED, CANCELLED, or UNCOMPLETED. The MODIFY/COMMAND command enables you to add, replace, or delete the qualifiers in the qualifier lists of these commands.

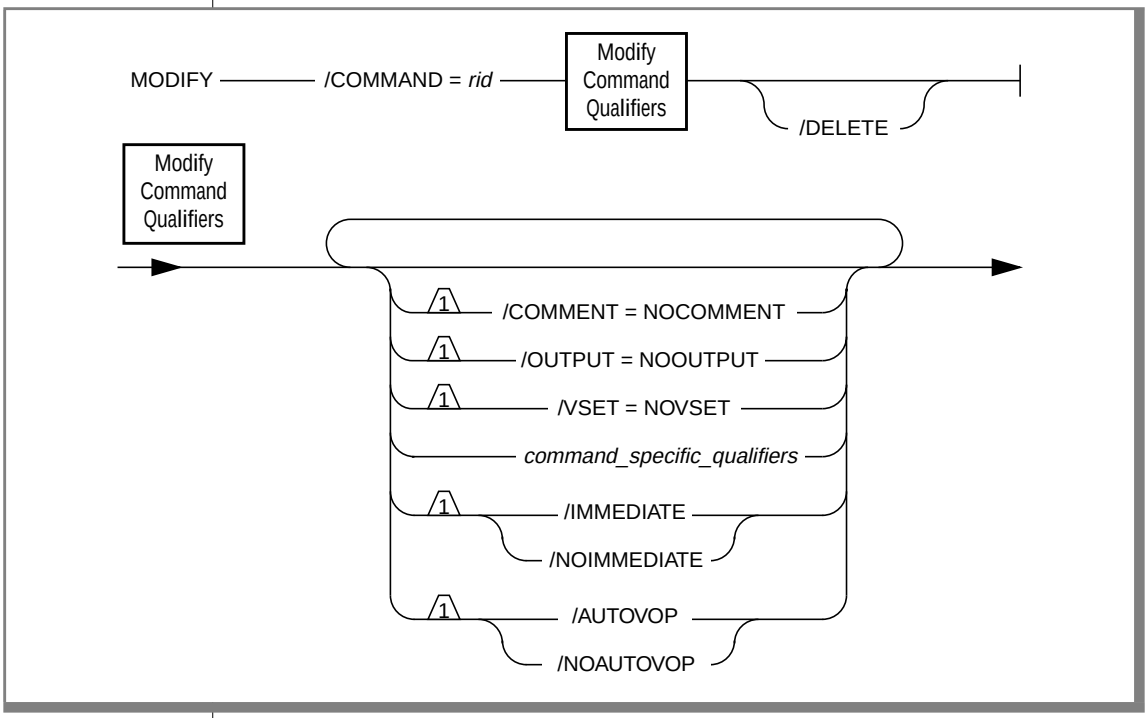
Modifying a request whose status is NEW simply updates its qualifier list with the specified changes. Modifying requests with any of the other permitted status values generates a new request with the updated qualifier list.

You can use the DELETE qualifier when you modify requests of FAILED, CANCELLED, or UNCOMPLETED status.

When you use the **onarchive** menu interface, to delete qualifiers from a command, simply blank out the field on the menu.

The following restrictions apply to the MODIFY / COMMAND command:

- When ON-Archive is running in OPERATOR privilege mode, a user must log in as **informix** or **root** to modify a request. Users **informix** and **root** can modify any other user's requests.
- When ON-Archive is running in GROUP privilege mode, users can only modify their own requests. The **super_archive** group users can modify any other user's requests.



AUTOVOP	specifies when you want to proceed to other tasks without waiting for a request to finish.
NOAUTOVOP	specifies the default setting for the MODIFY/COMMAND command. Allows you to override an AUTOVOP default setting, specified in a personal default file, to ensure that the MODIFY/COMMAND command functions as in earlier versions.
COMMAND	specifies the original request to modify. *rid* is a request in the catalog.
command-specific-qualifiers	The qualifiers permitted with this command depend on the type of request being modified. For example, when you modify an ARCHIVE request, only the qualifiers permitted with that command are valid.
COMMENT= NOCOMMENT	removes the comment from the specified request.
DELETE	used only with requests with the status FAILED, CANCELLED, or UNCOMPLETED. This qualifier deletes the original request.
IMMEDIATE	simplifies the process of creating and executing a request. The IMMEDIATE qualifier saves time. It allows you to create and execute a request using a single command. Displays the request ID associated with that request entry.
NOIMMEDIATE	specifies the default setting for the MODIFY/COMMAND command. Allows you to override an IMMEDIATE default setting, specified in a personal default file, to ensure that the MODIFY/COMMAND command functions as in earlier versions.
OUTPUT= NOOUTPUT	removes the OUTPUT qualifier from the specified request.
VSET=NOVSET	removes the VSET qualifier from the specified request.

See "The REMOVE/FAILED_REQUEST Command" on page 8-15 and "The REMOVE/REQUEST Command" on page 8-17 for more information on removing requests from the ON-Archive catalog. See "Groups of Qualifiers" on page 8-67 for more information on qualifiers that you could modify.

MODIFY/COMMAND Examples

The following example removes the comment from request 11.

```
Onarchive> MODIFY/COMMAND=11/COMMENT=NOCOMMENT
```

```
Request 00000011 modified
```

The following command creates a new request, using request 21 as a base, modifying the DBSPACESET qualifier in request 21. It also removes request 21 from the catalog.

```
Onarchive> MODIFY/COMMAND=21/DBSPACESET=*/DELETE
```

```
Request 00000034 registered in the catalog
Request 00000021 removed from the catalog
```

The following command removes the cancelled request 13 from the catalog:

```
Onarchive> MODIFY/COMMAND=13/DELETE
```

The following command negates the APART, LOG, and EXPIRY_DATE qualifiers for request 30:

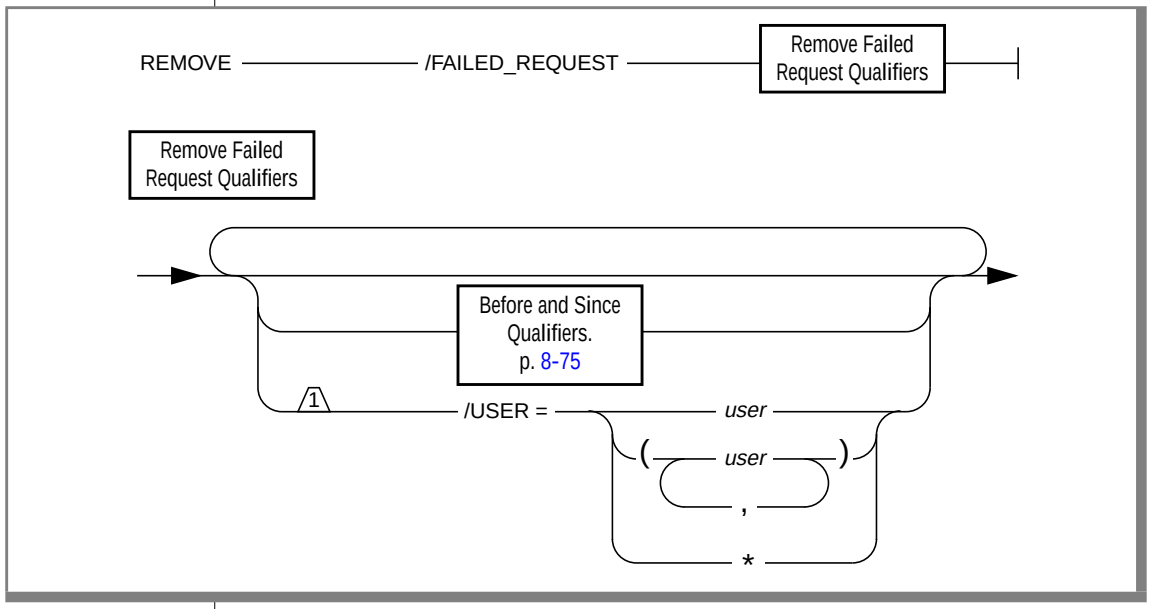
```
Onarchive> MODIFY/COMMAND=30/NOAPART/NOLOG/NOEXPIRY_DATE
```

The REMOVE/FAILED_REQUEST Command

The REMOVE / FAILED_REQUEST command removes all requests with a status of FAILED from the catalog. This command executes immediately. No new request id is generated.

The following restriction applies to the REMOVE / FAILED_REQUEST command:

- When ON-Archive is running in any privilege mode, and the user logs in as **informix**, **root**, or a member of the **super_archive** group, ON-Archive applies the command against all requests recorded in the catalog.
- When ON-Archive is running in any privilege mode, and the user is a regular ON-Archive user, ON-Archive applies the command only against those requests that the user owns.



FAILED_REQUEST	removes failed requests.
USER	specifies that the operation is performed only against requests that the specified user submitted.
	user specifies a valid user id.
	* specifies all users.

REMOVE/FAILED_REQUEST Examples

The following example removes all failed requests from the catalog:

```
Onarchive> REMOVE/FAILED_REQUEST
```

This example removes all failed requests from the catalog that were submitted before January 1, 1995:

```
Onarchive> REMOVE/FAILED_REQUEST/BEFORE=01-JAN-1995
```

The following example removes all failed requests from the catalog that were submitted 90 days ago or earlier:

```
Onarchive> REMOVE/FAILED_REQUEST/BEFORE=(TODAY - 90)
```

The REMOVE/REQUEST Command

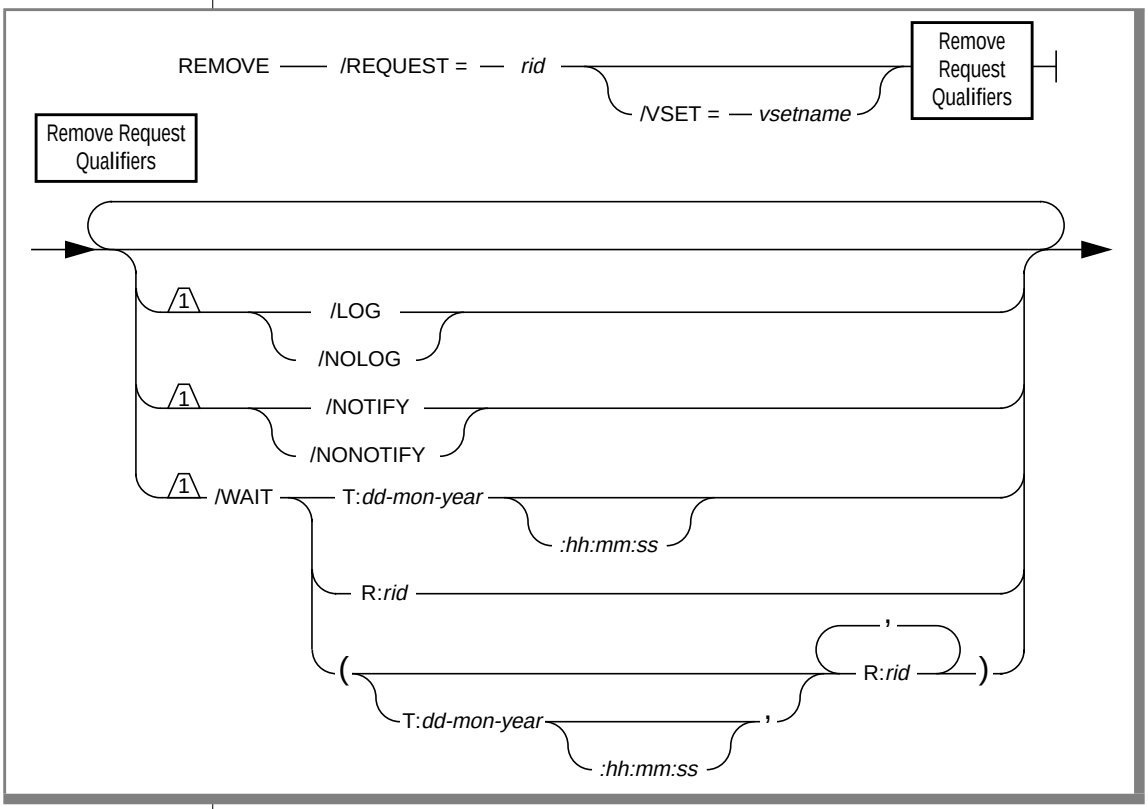
The REMOVE/REQUEST command removes the catalog entry for an entire save set (all data archived with the same request) from a specific vset. It only removes requests with a status of either CANCELLED or COMPLETED. (See "The CANCEL Command" on page 8-4 to cancel a command and see "The REMOVE/FAILED_REQUEST Command" on page 8-15 to remove requests that display other statuses.)

You typically use this command to reduce the number of copies of a specific save set on different vsets. The REMOVE/REQUEST command normally executes immediately without generating any requests. However, when you use the WAIT qualifier, it generates a request.

The REMOVE/REQUEST command does not physically delete save sets from tapes, it only changes the ON-Archive catalog. when the save set resides on a disk volume, REMOVE/REQUEST physically removes the save-set files from disk, in addition to modifying the ON-Archive catalog. See "Removing Save Sets to Reclaim Storage Space" on page 8-19 for a discussion of what you need to do with tape volumes after you use the REMOVE/REQUEST command.

The following restrictions apply to the REMOVE/REQUEST command:

- When ON-Archive is running in OPERATOR privilege mode, only users **informix** and **root** can remove requests. They can remove requests that any user creates.
- When ON-Archive is running in GROUP privilege mode, the **super_archive** group users can remove requests that any member of the **super_archive** group creates. Users **informix** and **root** can remove requests that any user creates.



REQUEST removes only the dbspace sets or log files that belong to the save set that the specified request creates.

rid specifies a request in the catalog.

VSET specifies the vset from which you remove the save set. VSET is not required when the status of the request is CANCELLED or the request is a RETRIEVE request.

vsetname specifies a valid vset.

LOG	specifies that ON-Archive must display save-set and request-id specifications for each save set and request id removed. When it cannot immediately execute the command because the WAIT qualifier was used, LOG writes file specifications to a log file in the directory where **onarchive** or **onautovop** was started. The log file is named **ARC***rid***.LOG**, where *rid* is the request id of the REMOVE/REQUEST command.
NOLOG	specifies that the save-set and request-id specifications are not displayed as they are removed. It is the default.
NOTIFY	specifies that after the request executes, the command sends electronic mail to the user who created the request.
NONOTIFY	specifies that the command sends no mail after the request executes. It is the default.
WAIT	specifies whether the request must wait for certain events to occur before it can start. When you specify more than one event, the request cannot start until all events complete. When one of the requests does not exist or has failed, a request waiting for this event also fails. See "Wait and Repeat Qualifiers" on page 8-82.

REMOVE/REQUEST Example

The following example removes the entire save set of request 21 from the catalog:

```
Onarchive> REMOVE/REQUEST=21/VSET=arc_test1
```

Removing Save Sets to Reclaim Storage Space

You can use the REMOVE/REQUEST command in the following situations to reclaim space that contains obsolete data:

- You did not specify an expiration date when you created an archive or backup request.
- You did not run **onautovop** to process the expiration dates.

When you remove a save set on a disk volume from the catalog, the archive or backup data also is removed automatically from the volume.

For sequential media, like tape, you cannot physically remove data from random locations. Thus, data stored on a tape volume is never deleted; it is overwritten. However, you can overwrite it only when you remove the information, corresponding to *all* the save sets on the tape, from the catalog. (Conversely, you can recatalog save sets stored on a tape volume when the information about them has been inadvertently deleted from the catalog. See "The CATALOG Command" on page 8-21 for information on how to recatalog save sets.)

The only way to reclaim space that is only partially used is to copy the save set still in use to another vset using the COPY / VSET command; then delete the original save set. See "The COPY / VSET Command" on page 8-24 for information on how to copy a save set.

Working with Volume Sets and Volumes

ON-Archive provides the following commands for working with volume sets (vsets) and volumes you defined previously using the DEFINE / VSET and DEFINE / VOLUME commands.

Command	Purpose
CATALOG	Creates a request that catalogs the contents of a volume.
COPY / VSET	Creates a request that copies the contents of one vset to another vset.
DELETE / USER	Removes the listed users from the access list for a vset.
DELETE / VOLUME	Removes a volume definition from the catalog.
DELETE / VSET	Removes a vset definition from the catalog.
EXECUTE / VSET	Provides a batch environment in where you can selectively group and execute requests.
LIST / VOLUME	Displays some or all of the volumes of a vset.

(1 of 2)

Command	Purpose
LIST / VSET	Displays information from the catalog for the listed vsets.
MODIFY / VOLUME	Changes a volume definition.
MODIFY / VSET	Changes a vset definition.

(2 of 2)

The CATALOG Command

The CATALOG command creates a request that, when executed, catalogs the contents of a volume. Manually cataloging a volume is required only in the following cases:

- You wish to retrieve data from a vset that another installation of ON-Archive created. In this case, the vset is said to be *imported*, and you cannot write on it or copy it to another vset. You can only retrieve data from it. See "What Is an Imported Volume Set?" on page 2-11.
- The ON-Archive catalog has been corrupted. You could catalog a volume again so that the catalog correctly lists the volume contents. See "Reconstructing the Catalog" on page 8-24.
- When the logical log fills and Universal Server activity stops, you must use **ondatartr** to back up the logical-log files. You must catalog the volumes that **ondatartr** created (and then copied to a regular vset) before you can use them. See "Performing Emergency Logical-Log File Backups" on page 6-13.
- When you perform a cold restore, you can salvage any logical-log files that were not backed up. You must catalog the volumes that **ondatartr** created (and then copied to a regular vset) before you can use them. See "Steps to Restore the Whole System" on page 7-22.

In all cases, ON-Archive scans the volume and rebuilds the ON-Archive catalog so that the data stored on the volume is retrievable.

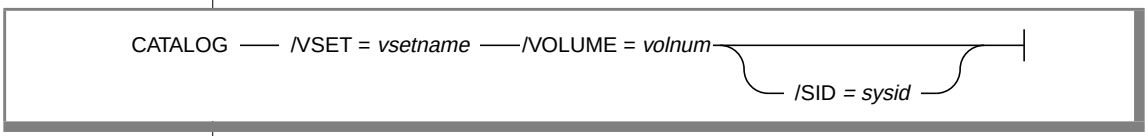
When the vset is defined as nonpermanent, ON-Archive prompts you to enter the device where the vset is mounted.

When recataloging, ON-Archive also reconstructs the requests that created save sets on the volume. When a request id already exists in the catalog and the text of the request and execution date are not the same as those of the request on the volume, the recataloged request is assigned a new request id. Associated with that new request is also a *former request id*, that is the request id as it was stored on the tape. You can see the former id when you use the FULL qualifier with the LIST / REQUEST command.

The file and save-set information is also recataloged, as well as archive event information.

The following restrictions apply to the CATALOG command:

- You must define the vset and the volume referred to by a CATALOG command before you issue the CATALOG command. See "Reconstructing the Catalog" on page 8-24 for more information.
- When ON-Archive is running in OPERATOR privilege mode, only users **root** and **informix** can use the CATALOG command.
- When ON-Archive is running in GROUP privilege mode, users in the **super_archive** group can use the CATALOG command.
- Users can only catalog vsets and volumes when they can read the vsets and volumes. See "Reconstructing the Catalog" on page 8-24 for more information.
- Users can only catalog disk volumes when they can read all the save sets on the volume they want to catalog. See "Reconstructing the Catalog" on page 8-24 for more information.



VSET specifies the vset you intend to catalog.

vsetname is the name of a vset in the ON-Archive catalog.

VOLUME specifies the volume you intend to catalog.

volnum is the number of the volume being cataloged. You can list the volume using the **ondatartr** command "The LIST/VOLUME Command" on page 8-36 prior to cataloging to obtain the volume number. ON-Archive assigns a new number to the volume as it is cataloged.

SID specifies the unique, user-defined *system identifier* for each installation of ON-Archive. This qualifier is mandatory for imported vsets. For example, when you catalog a vset from another Universal Server instance, you could use the server number of the database server the vset is from as the SID. You do not need a SID when you catalog vsets or volumes from the same ON-Archive system because of accidental deletion or corruption.

sysid specifies a system identifier within the range 1-999.

CATALOG Example

The following example catalogs the contents of vset **customers**, volume number 1.

```
Onarchive> CATALOG/VSET=customers/VOLUME=1
```

Reconstructing the Catalog

When you lose or corrupt the ON-Archive catalog, you can use the CATALOG command to rebuild it directly from the stored data. You could do this in the following situations:

- Accidental removal or corruption of the **sysmaster** database
- Accidental deletion of vsets or volumes

In both cases, the volumes contain data in them, but ON-Archive has no knowledge of them, so you cannot use them. You must perform the following steps so ON-Archive can recognize your volumes.

1. When your **sysmaster** database is gone, you need to rebuild it, as described in the *INFORMIX-Universal Server Administrator's Guide*.

Warning: *Do not rebuild* ***sysmaster*** *when it already exists! Rebuilding* ***sysmaster*** *drops the existing* ***sysmaster*** *database, including any existing information in the ON-Archive catalog.*

2. Use **ondatartr** to get the names of the vsets and number of the volumes. Use the LIST command for each volume.
3. Use **onarchive** to redefine the vsets and volumes you found with **ondatartr**. Use a dummy tape for **onarchive** to initialize when you define volumes, providing that tape volumes exist. Use pathnames that do not match those of existing volumes when disk volumes exist. Imported vsets do not require ON-Archive to actually initialize new volumes when you define them.
4. When you use disk volumes, use your operating-system **copy** command to move the volume-header file and save-set files into the new volume directory.
5. Finally, use the CATALOG command to catalog each volume.

See "Working with the Catalog" on page 8-64 for more information about repairing problems with the catalog.

The COPY/VSET Command

The COPY command creates a new request that, when executed, copies the contents of one vset (or volume) onto another vset. You must previously define both the source and the destination vsets.

The COPY command is a powerful storage-management tool. You can use it for the following tasks:

- Media refreshment

  Copy an entire vset into another with similar definition, then delete the old one.

- Reclaiming storage space

  Copy only unremoved and unexpired save sets from one vset into another with a similar definition, then delete the old one.

- Media conversion

  Copy a vset defined on one type of media into another vset defined on another type of media (for example, converting a volume from disk to tape).

- Creating an extra copy of a specific save set

  Copy a save set from one vset to another.

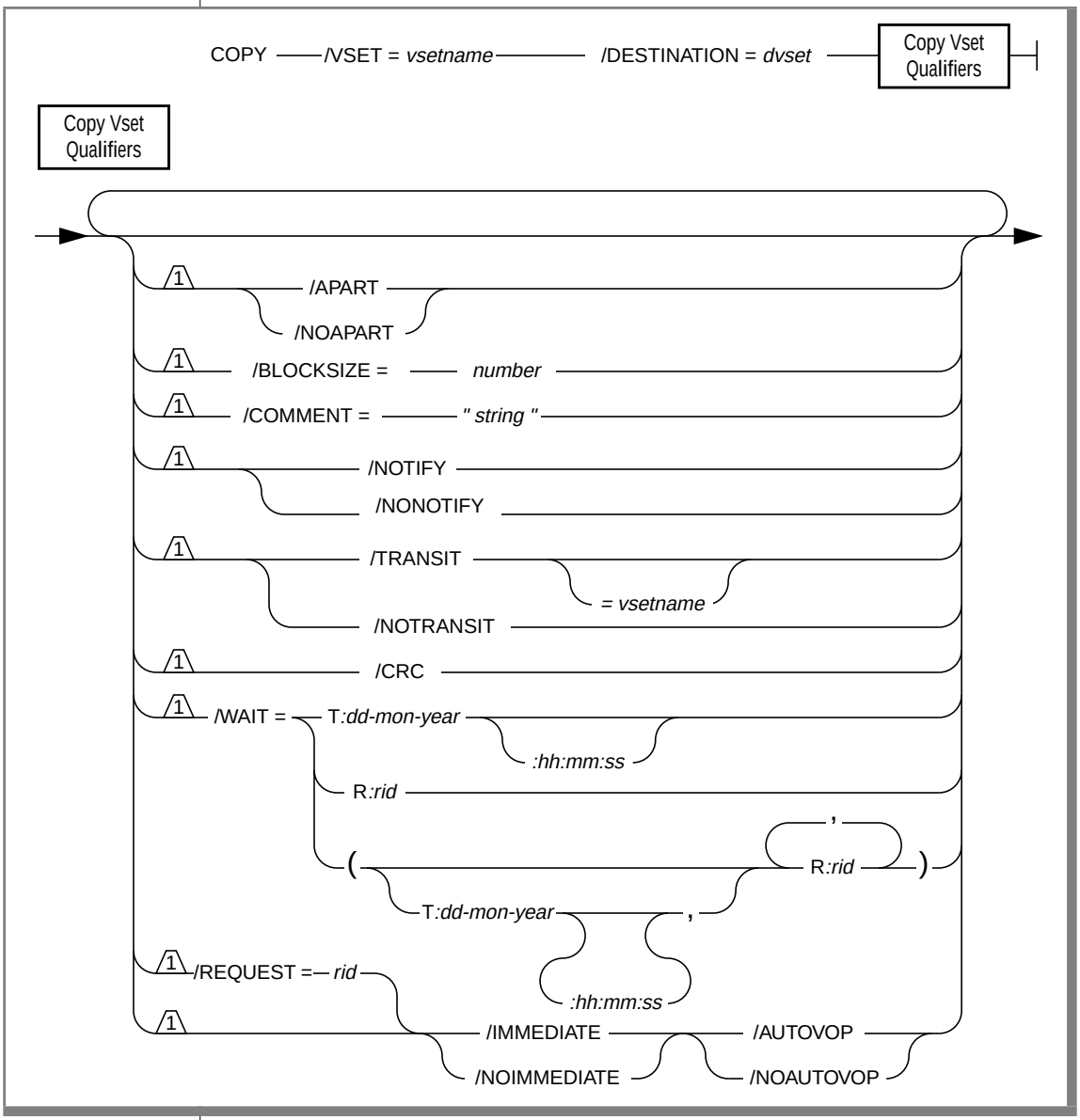
- Data separation

  Copy all save sets that belong to one user from one vset into another.

The following restrictions apply to the COPY command:

- To copy a save set, you must retain permission to access the volume where the save set resides, and to both source and destination vsets.
- The save sets selected from the input vset must not already exist in the destination vset.
- When ON-Archive is running in OPERATOR privilege mode, users **root** and **informix** can copy save sets, even when they did not create the request that created the save set.
- When ON-Archive is running in GROUP privilege mode, users in the **super_archive** group can copy save sets, even when they did not create the request that created the save set.
- Only **informix** or **root** can copy an entire vset, regardless of the privilege mode in which ON-Archive is running.
- When the COPY request copies an entire vset and, therefore, generates child requests to copy the individual save sets, the child requests do not execute automatically; you must execute them. See "REQUEST" page 8-28 for more information.

Once you create the requests, restrictions also apply to their execution. See "The EXECUTE / VSET Command" on page 8-33.



APART — specifies that you must keep the save set on a volume separate from other save sets (that is, you write the save set to a blank volume and you do not write any other save sets to that volume). Whenever you write to a remote tape device, you must use the APART qualifier.

NOAPART — indicates that the save set can reside on volumes with other save sets. It is the default.

AUTOVOP — specifies when you want to proceed to other tasks without waiting for a request to finish.

NOAUTOVOP — specifies the default setting for the COPY/VSET and COPY/VSET/REQUEST commands. Allows you to override an AUTOVOP default setting, specified in a personal default file, to ensure that the COPY/VSET and COPY/VSET/REQUEST commands function as in earlier versions.

BLOCKSIZE — specifies the block size in bytes when writing to tape volumes or, with NB_DISK_SPACE_EXTENT, the size of disk space allocations when writing to disk. See "Disk-Space Allocation During Concurrent Operations" on page 2-22 of this manual.

The internal default value is 64 kilobytes as specified in the operator default file. See "Where Qualifier Default Values Are Specified" on page 4-13 of this manual.

number — is an integer specifying the block size in bytes. The valid block-size range is from 8197 to 65,024.

COMMENT — specifies a text string containing a comment about the copy operation. The comment is stored with the save set in the ON-Archive catalog.

string — is a text string. It cannot exceed 80 characters.

CRC	instructs ON-Archive to add a cyclic-redundancy check (CRC) at the end of each save-set block.
DESTINATION	specifies the name of a vset to copy to. *dvset* is the name of an existing vset.
IMMEDIATE	simplifies the process of creating and executing a request. The IMMEDIATE qualifier saves time. It allows you to create and execute a request using a single command. Displays the request ID associated with that request entry.
NOIMMEDIATE	specifies the default setting for the COPY/VSET and COPY/VSET/REQUEST commands. Allows you to override an IMMEDIATE default setting, specified in a personal default file, to ensure that the COPY/VSET and COPY/VSET/REQUEST commands function as in earlier versions.
NOTIFY	specifies that after the request executes, the command sends electronic mail to the user who created the request.
NONOTIFY	specifies that the command sends no mail after the request executes. It is the default.
REQUEST	specifies that you can copy only save sets that the specified request creates. *rid* is an existing request id. When you do not use the REQUEST qualifier to specify particular save sets, the command copies the entire vset to the destination vset. To do this, ON-Archive generates a new child request for each save set in the vset to copy it to the destination vset. You must then execute those requests.

TRANSIT — controls whether you must use a transit vset. See "What Is a Transit Volume Set?" on page 2-11 of this manual.

When you use TRANSIT without any parameters, ON-Archive selects a transit vset. It searches first through the user's transit vsets for an available transit vset. When transit vsets are unavailable, ON-Archive then searches the system-transit vsets. See the CLASS qualifier for the DEFINE/VSET command on page 4-34 of this manual for more information on system and user vsets.

You need a transit vset for COPY when the source and destination vset require the same device type, and only one device of that type is available.

vsetname is the name of a transit vset.

NOTRANSIT — specifies that you must not use a transit vset. It is the default.

VSET — specifies the name of a vset to copy from.

vsetname is the name of an existing vset.

WAIT — controls whether a request must wait for certain events to occur before it can start. See "Wait and Repeat Qualifiers" on page 8-82 of this manual.

COPY/VSET Example

The following example creates a request to copy vset **may93** to vset **may93bkp**:

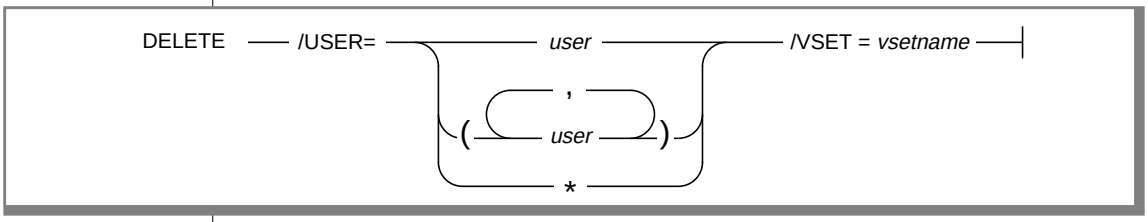
```
Onarchive> COPY/VSET=may93/DESTINATION=may93bkp

Request 00000041 registered in the catalog.
```

The DELETE/USER Command

The DELETE / USER command removes a user or users from the access list for a vset. To add users to an access list, use MODIFY / VSET.

The following restrictions apply to the DELETE / USER command:

- You can use the DELETE / USER command only with USER vsets.
- When ON-Archive is running in OPERATOR privilege mode, only users **informix** and **root** can delete a user from an access-control list.
- When ON-Archive is running in GROUP privilege mode, for USER vsets the user must reside on the access-control list of the vset.



USER removes specified users from the access-control list. The parameter * removes all users from the list.

user is a user id in the access-control list.

VSET specifies a vset.

vsetname is the name of a vset in the ON-Archive catalog.

DELETE/USER Examples

The following example removes the user **pat** from the access list of the **test** vset:

```
Onarchive> DELETE/USER=pat/VSET=test

  1 user(s) removed from access list for test
```

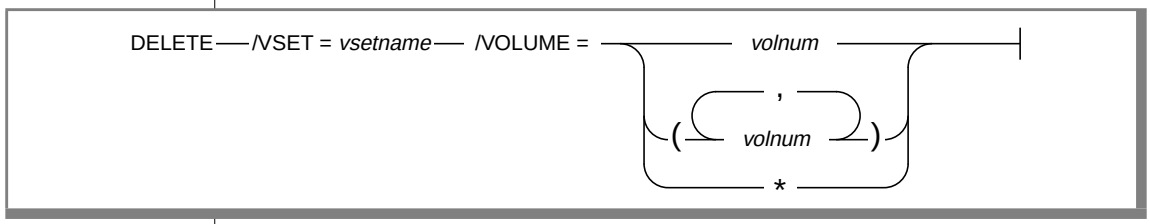
The DELETE/VOLUME Command

The DELETE/VOLUME command removes a volume definition from the ON-Archive catalog. When you delete the definition of a volume, it is as though it no longer exists. Consequently, you can no longer archive the data, or back up the data to the media, until it has been redefined as a new volume.

When you delete a disk volume, the DELETE/VOLUME command removes the volume header from the volume directory.

The following restrictions apply to the DELETE/VOLUME command:

- The volume being deleted must not contain any save sets that reside in the catalog. (In other words, the volume must be empty.) When the volume is not empty, use the REMOVE/REQUEST command to remove the save sets from the catalog.
- You can delete a volume only when the vset to which it belongs and the volume possess delete permission (D) granted. When you do not grant delete permission to the volume, use the command MODIFY/VOLUME to change its protection. When the vset does not have delete permission, use MODIFY/VSET.
- When ON-Archive is running in OPERATOR privilege mode, only users **informix** and **root** can delete volumes.
- When ON-Archive is running in GROUP privilege mode, for SYSTEM vsets, only members of the **super_archive** group can delete volumes. For USER vsets, the user must also reside on the access-control list of the vset to delete volumes.



VSET specifies a vset.

vsetname is the name of a vset in the ON-Archive catalog.

VOLUME specifies the volumes to delete. The parameter * means delete all volumes defined for this vset, but you keep the vset definition.

volnum is the volume number. You could list volumes using "The LIST/VOLUME Command" on page 8-36 to get the volume number.

DELETE/VOLUME Example

The following example removes volume 1 from the **test** vset:

```
Onarchive> DELETE/VSET=test/VOLUME=1

1 volume(s) deleted
```

The DELETE/VSET Command

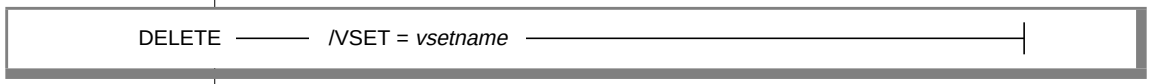
The DELETE/VSET command removes a vset definition from the ON-Archive catalog. When you delete the definition of a vset, it is as though it no longer exists. Consequently, you can no longer retrieve data from it. Also, you can no longer archive or back up data to the media until you redefined it as new volumes in a new vset.

The DELETE/VSET command does not erase the data stored on the archive media. When you accidentally issue a DELETE/VSET command, you can use the CATALOG command to restore the catalog. It reads the content description back from the physical media.

The following restrictions apply to the DELETE/VSET command:

- You cannot delete a vset when a request resides in the catalog that can write to the vset, when executed.
- You can delete a vset definition only when it has delete access (PROTECTION = D) granted and when you granted delete permission to all its volume definitions.

- When ON-Archive is running in OPERATOR privilege mode, only users **informix** and **root** can delete.
- When ON-Archive is running in GROUP privilege mode, for SYSTEM vsets, only members of the **super_archive** group can delete vsets. For USER vsets, the user must also reside on the access-control list of the vset to delete the vset.



VSET specifies a vset.

vsetname is the name of the vset to delete from the ON-Archive catalog.

DELETE/VSET Example

The following example removes the definition of the **test** vset:

```
Onarchive> DELETE/VSET=test

2 volume(s) deleted
All users removed from access list for TEST
Vset TEST deleted
```

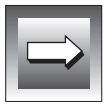
The EXECUTE/VSET Command

The EXECUTE/VSET command executes all requests that name the specified vset as their destination.

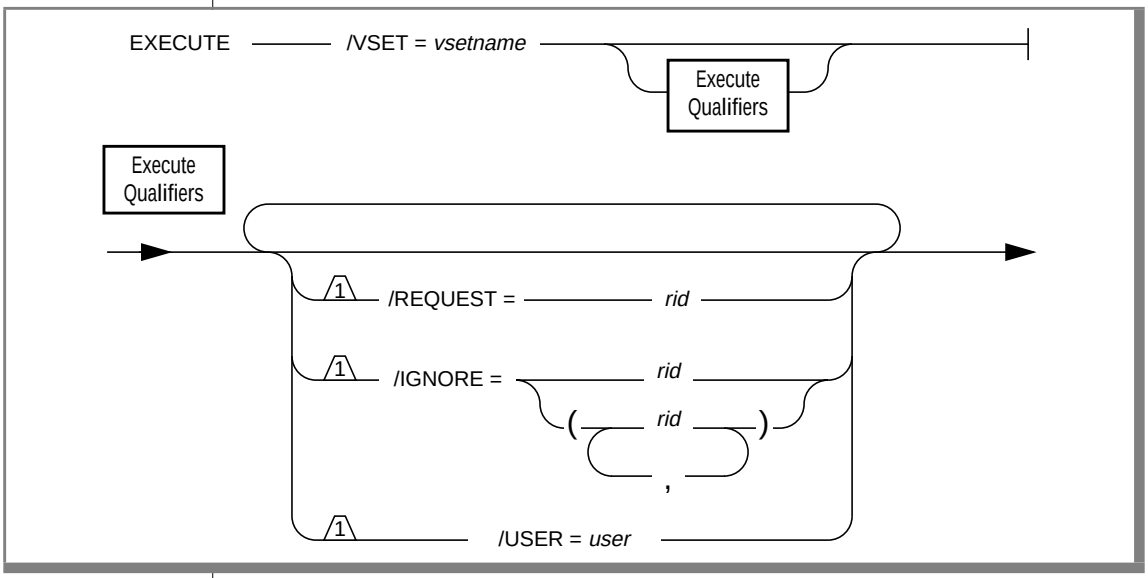
When you define the specified vset as nonpermanently mounted, ON-Archive issues one mount operation. It then executes all the requests that match the specified qualifiers and unmounts the device.

The following restrictions apply to the EXECUTE / VSET command:

- When ON-Archive is running in OPERATOR privilege mode, the user must log in **informix** or **root**. They can execute any request.
- When ON-Archive is running in GROUP privilege mode, the user must log in as a member of the **super_archive** group. The user can then execute requests for any user in the **super_archive** group.
- You must grant users appropriate privileges on the vset and volume the request is reading or writing to for the request to execute successfully.



Important: *Do not confuse the EXECUTE/VSET command that the EXECUTE command. Use the EXECUTE command to execute a request that you identify using its request id. See* *"The EXECUTE Command" on page 4-27* *for more information.*



VSET specifies the destination volume set from which you select requests for execution. You can select only ARCHIVE, BACKUP, and child-copy requests for execution.

vsetname is the name of the regular vset specified for requests in the ON-Archive catalog. You must not use a transit vset that was specified during the submission of the request.

REQUEST ON-Archive starts the execution of the specified request. After you complete the request, ON-Archive returns to the operating-system prompt where the EXECUTE command was issued.

When you omit the REQUEST qualifier, the command considers all requests for the vset specified for execution.

rid specifies the request id of a request currently in the catalog.

IGNORE specifies a request or a list of requests that you cannot consider for execution.

rid specifies the request id of a request currently in the catalog.

USER specifies that you must consider requests, that the specified user submits, for execution.

user is a valid operating-system user id.

EXECUTE/VSET Examples

This command executes all requests that use the final destination called **test**:

```
Onarchive> EXECUTE/VSET=test
```

The following example executes all requests that the user **smith** submits, except requests 126 and 181, that use the final destination vset called **accounting**:

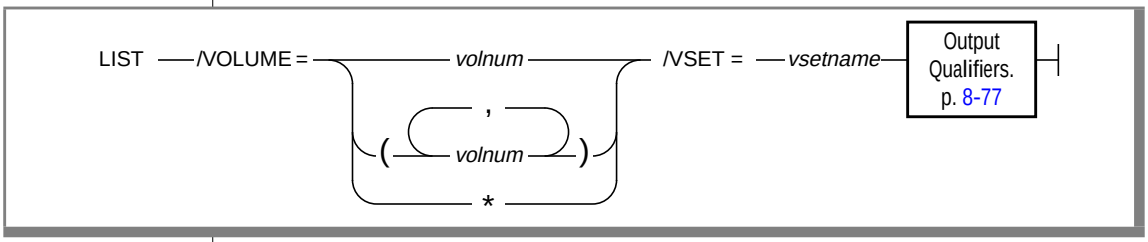
```
Onarchive>
EXECUTE/VSET=accounting/USER=smith/IGNORE=(126,181)
```

The LIST/VOLUME Command

The LIST / VOLUME command displays some or all of the volumes of a specific vset.

The following restrictions apply to the LIST / VOLUME command:

- When ON-Archive is running in OPERATOR mode, only **informix** or **root** can list volume information. They can list information on any volume.
- When ON-Archive is running in GROUP privilege mode, only **super_user** group users can display information about USER vsets providing that they are not included in the vset access-control list. Otherwise, users must log in as **informix** or **root** to view USER vsets when they are not included in the vset access-control list.



VOLUME specifies the volumes designated for display.

volnum is a valid volume number in the ON-Archive catalog.

* specifies all volume numbers in the ON-Archive catalog.

VSET specifies the vset to which the volume(s) belong.

vsetname is a valid vset.

LIST/VOLUME Example

The following example shows a brief listing of all the volumes in the vset called **temp**:

```
Onarchive> LIST/VSET=TEMP/VOLUME=*/BRIEF
```

```
Volume set name: TEMP

Volume No   Nb Save set   Nb physical save set   Volume label   Prot   Max_Space   Used_Space   F
1           2             2                      NR2351         RWD    500         499          Y
2           4             4                      NY6513         RWD    6000        5999         Y
3           2             2                      TRE965         RWD    12000       180          N
```

The following example shows a full listing of the volume in the vset called **temp** (it only uses one volume; the previous example used three):

```
Onarchive> LIST/VSET=TEMP/VOLUME=*/FULL
```

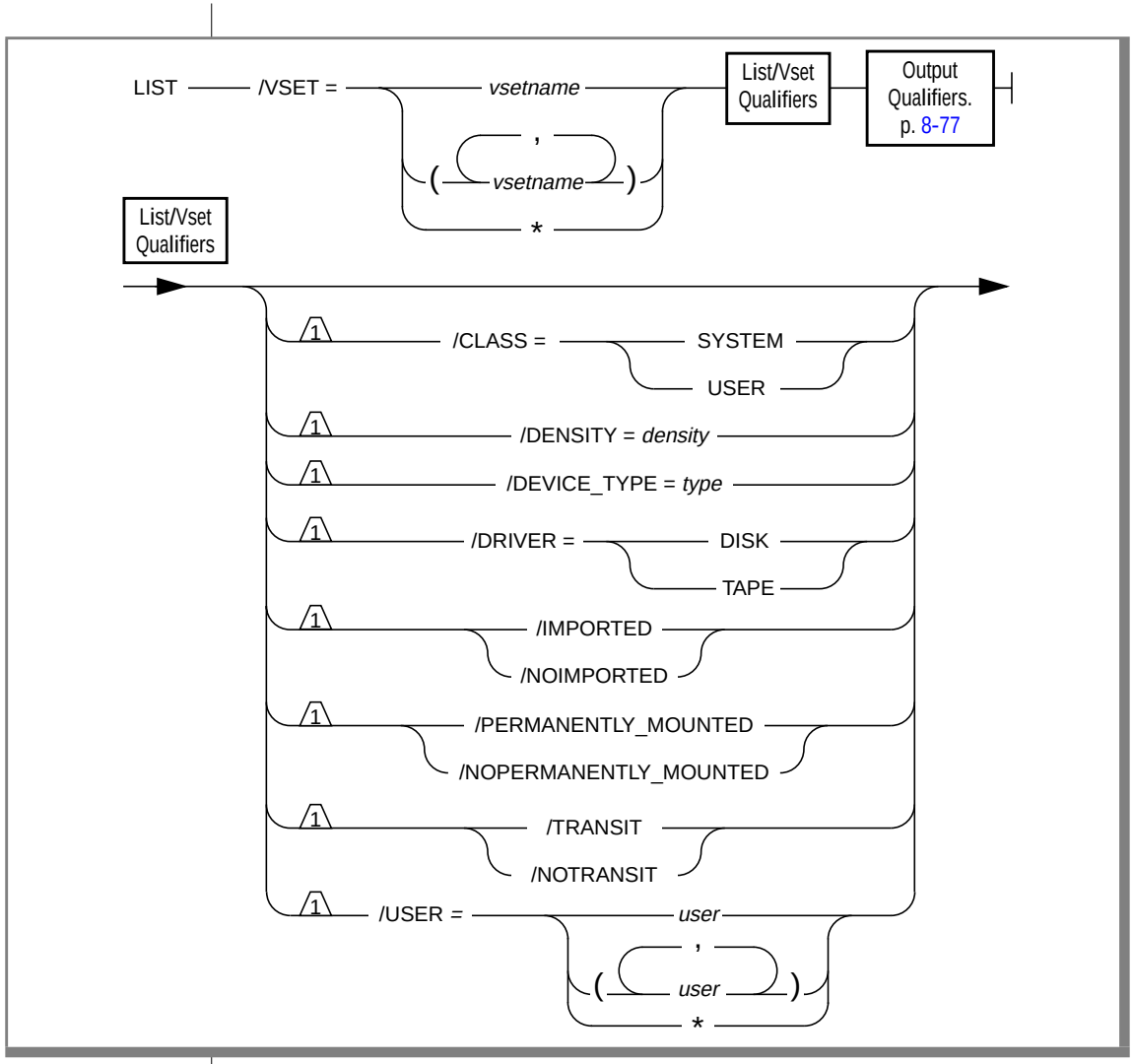
```
Volume name: TEMP

Volume No:      1        Volume label: NR2351
Nb. save set:   2        Nb. phys save set: 2
Protection:     RWD
Max Space:      5000     Used space :  499      Full: No
Virtual:        /usr/arc_vset/temp/vol1
Parameters:
Comment:
```

The LIST/VSET Command

The LIST / VSET command displays information about vsets defined in the ON-Archive catalog. SYSTEM vsets are visible to all users, but USER vsets are only visible to the users who gain access.

The following restrictions apply to LIST / VSET:

- When ON-Archive is running in OPERATOR mode, users must log in as **informix** or **root** to list vset information. They can list information on any vsets.
- When ON-Archive is running in GROUP privilege mode, users must log in as part of the **super_user** group to display information about USER vsets when they are not included in the vset access-control list. Otherwise, the user must log in as **root** or **informix** to view USER vsets when the user does not reside in the vset access-control list.


LIST
/VSET =
vsetname
,
vsetname
*
List/Vset Qualifiers
Output Qualifiers. p. 8-77
List/Vset Qualifiers
1
/CLASS =
SYSTEM
USER
/DENSITY = density
/DEVICE_TYPE = type
/DRIVER =
DISK
TAPE
/IMPORTED
/NOIMPORTED
/PERMANENTLY_MOUNTED
/NOPERMANENTLY_MOUNTED
/TRANSIT
/NOTRANSIT
/USER =
user
,
user
*

VSET	specifies the vsets to display. *vsetname* is a valid vset.
CLASS	displays only the vsets that belong to the specified class (SYSTEM or USER). For a description of the CLASS qualifier, see "Access-Control Lists" on page 2-26.
DENSITY	lists only the vsets defined with the specified density. *density* is the same density used to define a vset.
DEVICE_TYPE	displays only the vsets defined for the specified device type. *type* is a logical device listed in the ON-Archive configuration file, **config.arc**.
DRIVER	lists only the vsets defined for the specified driver (DISK or TAPE).
IMPORTED	lists only the vsets that you import.
NOIMPORTED	lists only the vsets that you do not import. When you omit these qualifiers, the command displays all vsets (imported and not imported).
PERMANENTLY_ MOUNTED	displays only the permanently mounted vsets.
NOPERMANENTLY_ MOUNTED	displays only the vsets not permanently mounted. When you omit these qualifiers, the command displays all vsets (permanently mounted or not).
TRANSIT	displays only the vsets defined as transit vsets.

NOTRANSIT — displays only the vsets not defined as transit vsets.

When you omit these qualifiers, the command displays all vsets (transit or not).

USER — displays only the vsets that the specified user can access. When the parameter is *, the command includes all users.

user is a valid user id.

LIST/VSET Examples

The following example shows a brief description of all defined vsets:

```
Onarchive> LIST/VSET=*/BRIEF
```

```
Volume set nameClass Owner NodeDriver DeviceAccTPIOVol
------------------- ---------------- ----------------
MISC    SYSTEMsparcDISK/usr45 NYN     Y       2
TEMP    SYSTEMsparcDISK/usr30 NPN     Y       3
TRANSITVSETSYSTEMsparcDISK/usr10 YYN Y       1
OFFLINESYSTEMsparcTAPE/dev/rst070 NNNY       6
ACCOUNTINGUSERsparcDISK/usr65 NNN     Y       1
```

This example shows a full description of a defined vset:

```
Onarchive> LIST/VSET=MISC/FULL
```

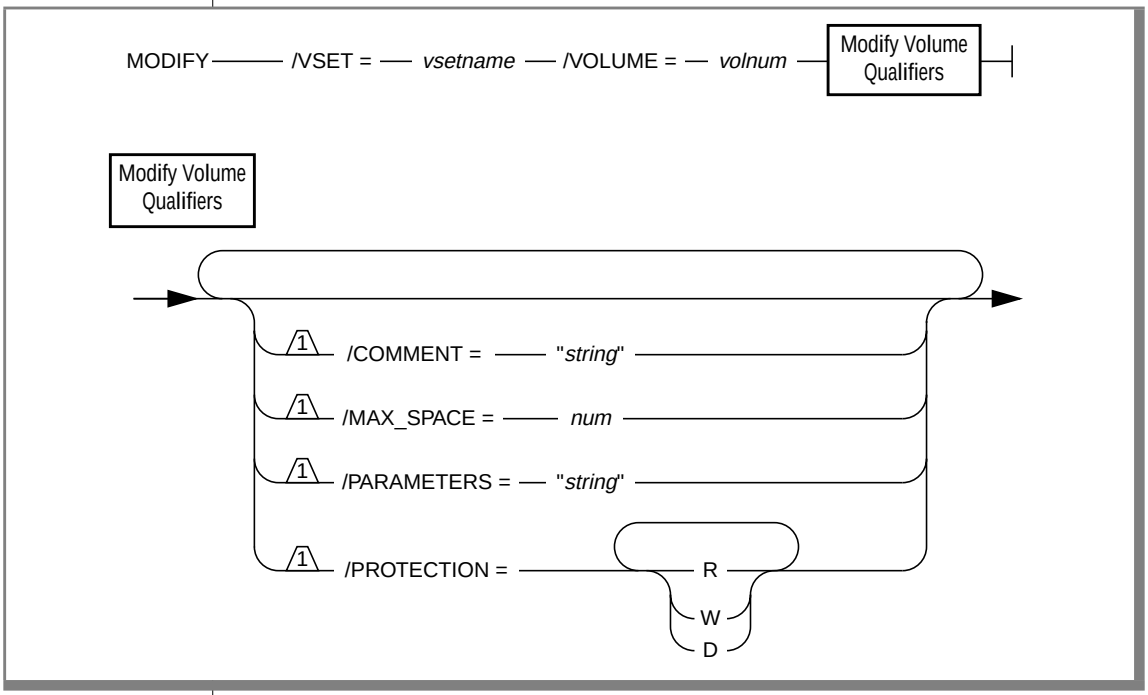
```
Vset name: MISCClass: SYSTEMVolumes: 2
Owner Node: sparcDevice: eagle
Driver: DISKDensity: OTransit  : NOPermanent : YES
Accessibility : 45Protection : RWDImported : NO
Parameter(s):Onsite: YES
Location:
Comment: THIS VSET IS USED FOR MISCELLANEOUS ARCHIVING
```

The MODIFY/VOLUME Command

The MODIFY / VOLUME command changes a volume definition. You can change only attributes shown in the command syntax. You can modify a volume only when it is not locked.

The following restrictions apply to the MODIFY / VOLUME command:

- When ON-Archive is running in OPERATOR privilege mode, the user must log in as **informix** or **root** to modify vsets.
- When ON-Archive is running in GROUP privilege mode and the user is part of the **super_archive** group, the user can modify SYSTEM vsets. To modify volumes in USER vsets, the user must reside in the vset access-control list.



VSET — indicates the vset name of the vset designated for modification.

vsetname is a valid vset name.

VOLUME — specifies the volume number of the volume definition designated for modification.

volnum is a valid volume number.

COMMENT — specifies a comment for this request.

string is a text string containing a comment about the volume. The string cannot exceed 80 characters in length.

MAX_SPACE — changes the maximum volume space to the specified value. The space the volume currently uses can exceed the maximum space.

num is a value between 0 and 99999999 (in blocks of 512 bytes).

PARAMETERS — is reserved for future use.

PROTECTION — changes the volume protection to the specified value. See "Protection Qualifier" on page 8-81.

MODIFY/VOLUME Example

The following example changes the comment associated with volume number 1 of vset **arc_test1**:

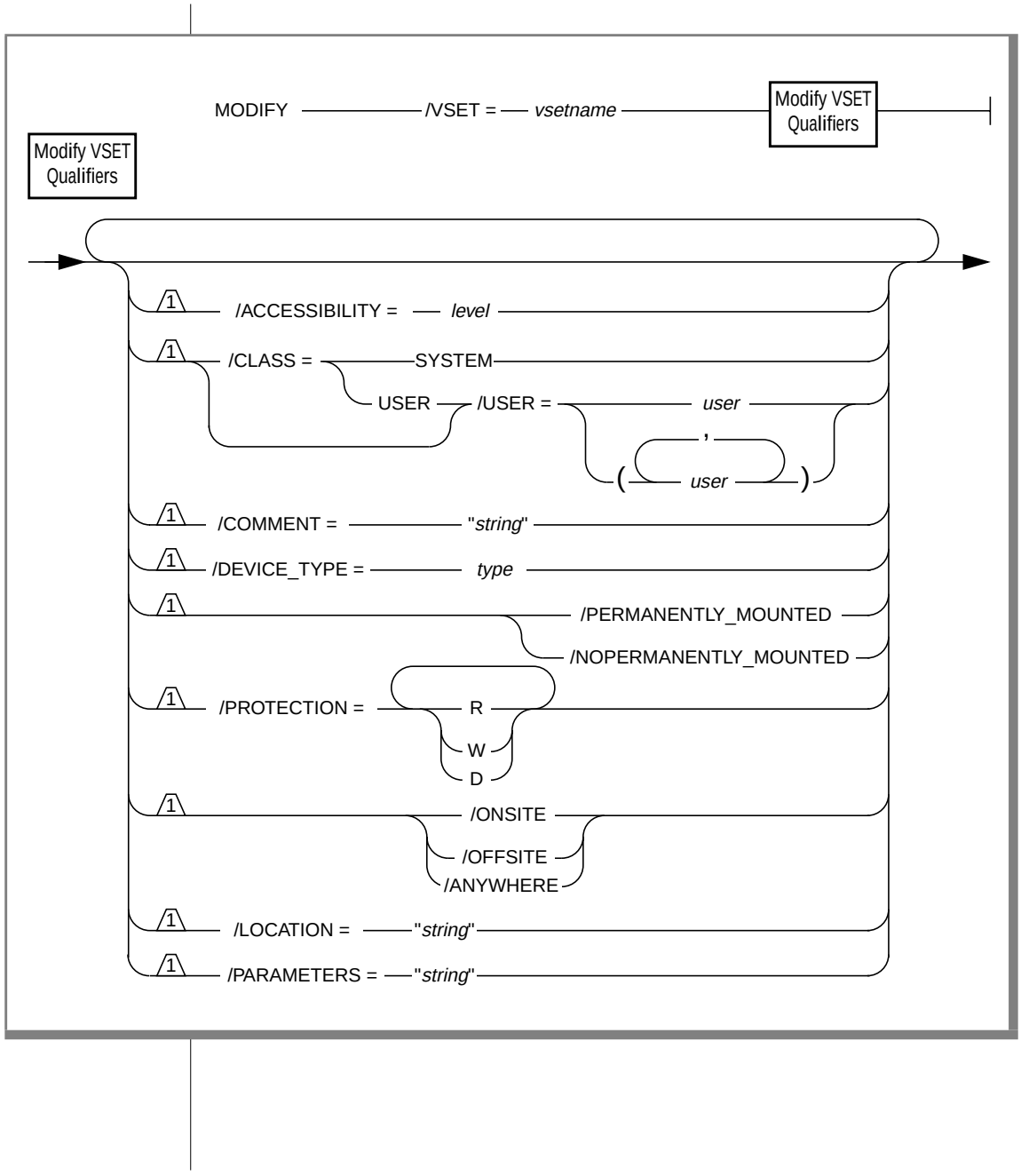
```
Onarchive> MODIFY/VOLUME=1/VSET=arc_test1/COMMENT="vol 1 of vset arc_test1"

Volume 1 from ARC_TEST1 modified
```

The MODIFY/VSET Command

The MODIFY / VSET command changes a vset definition. You can change only the attributes shown in the command syntax. You can modify a vset only when it is not locked.

The following restrictions apply to MODIFY / VSET:

- When ON-Archive is running in OPERATOR privilege mode, the user must log in as **informix** or **root** to modify vsets.
- When ON-Archive is running in GROUP privilege mode, the user must log in as part of the **super_archive** group to modify vsets. The user can modify SYSTEM vsets. To modify USER vsets, the user must reside in the vset access-control list.


MODIFY
/VSET =
vsetname
Modify VSET Qualifiers
Modify VSET Qualifiers
/ACCESSIBILITY =
level
/CLASS =
SYSTEM
USER
/USER =
user
,
user
/COMMENT =
"string"
/DEVICE_TYPE =
type
/PERMANENTLY_MOUNTED
/NOPERMANENTLY_MOUNTED
/PROTECTION =
R
W
D
/ONSITE
/OFFSITE
/ANYWHERE
/LOCATION =
"string"
/PARAMETERS =
"string"

VSET	indicates the vset you modify. *vsetname* is a valid vset name.
ACCESSIBILITY	changes the accessibility of a vset. *level* is an integer from 0 to 99 specifying the accessibility of the vset.
CLASS	changes the class of a vset (SYSTEM or USER). For a description of the CLASS qualifier, see "Access-Control Lists" on page 2-26 and the CLASS qualifier for "The DEFINE/VSET Command" on page 4-31.
USER	indicates new users that you add to the vset access-control list for user vsets. To remove a user from the access-control list, use "The DELETE/USER Command" on page 8-30. *user* is a valid operating-system user id.
COMMENT	replaces the comment with the specified text. To erase the old comment, enter an empty string (" ") as parameter. *string* cannot exceed 80 characters in length.
DEVICE_TYPE	changes the vset device type. You must define the device type as a logical device on a DEVICE parameter in the ON-Archive configuration file. *type* is the logical device type for the vset.
PERMANENTLY_MOUNTED	specifies that you change the vset definition to permanently mounted.
NOPERMANENTLY_MOUNTED	indicates that you must not consider the vset as permanently mounted.
PROTECTION	changes the vset protection. See "Protection Qualifier" on page 8-81.

ONSITE	specifies that a vset resides on-site and thus physically accessible to operators. When the site status changes to ONSITE, the command immediately notifies the operator of any request waiting for that vset.
OFFSITE	specifies that ON-Archive does not consider the vset when it needs to select a vset. When you execute a request that specifies an off-site vset, the command displays an error and does not execute the request, but the status remains NEW.
ANYWHERE	specifies that whenever ON-Archive needs a volume from the vset, it asks the operator to check whether the volume resides on-site or not.
LOCATION	changes the vset location to the specified value.
string	is a text string describing the location of the vset. The string cannot exceed 80 characters in length.
PARAMETERS	is reserved for future use.

MODIFY/VSET Example

The following example changes the comment associated with vset **arc_test1**:

```
Onarchive> MODIFY/VSET=arc_test1/COMMENT="vset for mis dept"

Vset ARC_TEST1 modified
```

Changing Storage Access Permissions

To gain access to a volume or vset without write access, **informix** or **root** must modify the protection using the MODIFY/VSET or MODIFY/VOLUME command and the PROTECTION qualifier.

Working with Dbspace Sets

ON-Archive provides the following commands for working with dbspace sets that you previously defined with the DEFINE / DBSPACESET command:

Command	Purpose
DELETE / DBSPACESET	Deletes a dbspace-set definition from the catalog.
LIST / DATABASE	Displays the tables in the database, the dbspace and blobspaces (if any) where the tables reside, and the name of the dbspace set to which the tables belong.
LIST / DBSPACESET	Displays information for either a specific dbspace set or for all dbspace sets defined in the catalog.

See "The DEFINE / DBSPACESET Command" on page 5-5 for information on how to create a dbspace set.

The DELETE/DBSPACESET Command

The DELETE / DBSPACESET command deletes a dbspace-set definition from the ON-Archive catalog. When you delete the definition of a dbspace set, it is as though it no longer exists. Consequently, you can no longer use it to archive data.

The following restrictions apply to the DELETE / DBSPACESET command:

- You cannot delete a dbspace set when it is used to archive data. You can only delete it when it is not used—providing that no requests in the catalog reference it.
- You cannot delete a dbspace set when a request in the catalog can use the dbspace set.
- When ON-Archive is running in OPERATOR privilege mode, only users **informix** and **root** can delete dbspace sets.
- When ON-Archive is running in GROUP privilege mode, only users in the **super_archive** group can delete dbspace sets.

DBSPACESET specifies a dbspace set designated for deletion.

dbspsetname is the name of an existing dbspace set. "The LIST / VOLUME Command" on page 8-36 allows you to see the dbspace sets defined in the catalog.

DELETE/DBSPACESET Example

The following example deletes a dbspace set called **freq_use**:

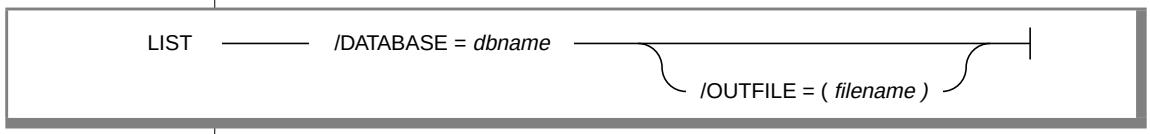
```
Onarchive> DELETE/DBSPACESET=freq_use
```

See "The DEFINE / DBSPACESET Command" on page 5-5 for information on defining a dbspace set.

The LIST/DATABASE Command

The LIST / DATABASE command displays the tables for a database, the dbspaces and blobspaces where the tables reside, and the dbspace set where the dbspaces and blobspaces reside.

The LIST / DATABASE command has no restrictions.



DATABASE specifies a database.

dbname is the name of a database that Universal Server manages.

OUTFILE specifies that you must send the output to a disk file instead of the screen. You cannot use the OUTFILE qualifier from the menu interface.

filename specifies the pathname and filename of the output file. You can use an absolute pathname, or a pathname relative to the directory where **onarchive** (or **onautovop**) was started. An absolute pathname must begin with a slash (/) and a relative pathname must *not* begin with a slash (/).

LIST/DATABASE Example

The following example lists the location of the tables in the **stores7** database:

```
Onarchive> list/database=stores7
```

```
DATABASE    TABLE          DBSPACE SET   DBSPACE/BLOBSPACE
------------------------------------------------------------
stores7     customer       crit_set      rootdbs
stores7     orders         crit_set      rootdbs
stores7     manufact       crit_set      rootdbs
stores7     stock          crit_set      rootdbs
stores7     items          crit_set      rootdbs
stores7     state          crit_set      rootdbs
stores7     call_type      crit_set      rootdbs
stores7     cust_calls     crit_set      rootdbs
stores7     catalog        crit_set      rootdbs
```

The LIST/DBSPACESET Command

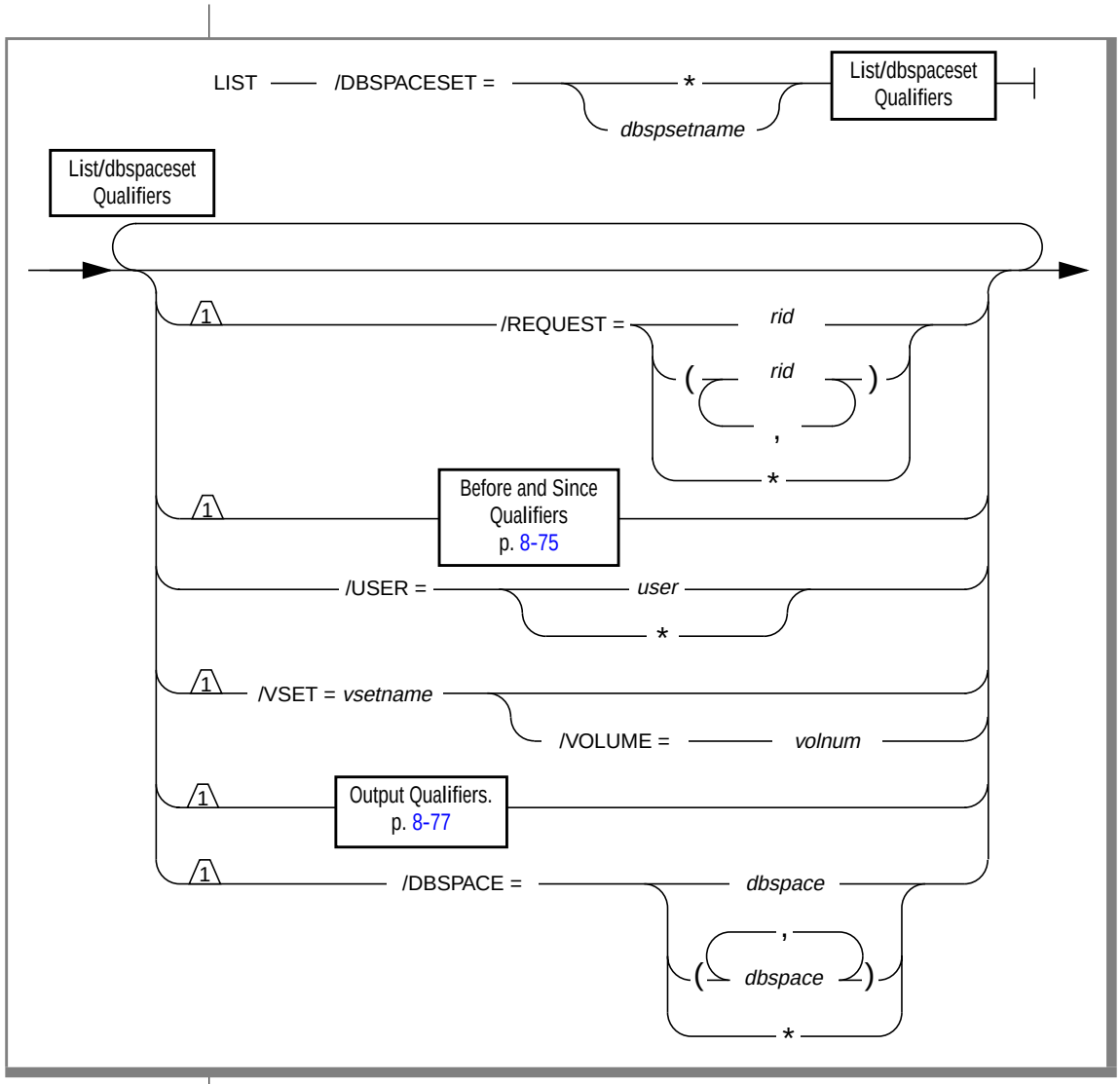
The LIST / DBSPACESET command displays information for either a specific dbspace set or for all dbspace sets.

When you specify the BRIEF output qualifier, ON-Archive prints only a one-line summary. When you specify the FULL qualifier, ON-Archive displays information for each archive request that you specify and identifies the vset and volume to which it wrote each dbspace. See "Output Qualifiers" on page 8-77 for information on the output qualifiers.

The LIST / DBSPACESET command lists * as a dbspace-set name because you can specify * as the dbspace-set name to indicate all dbspaces when you create an archive. When the output from the LIST / DBSPACESET command lists * as the dbspace-set name it indicates that the set consists of all dbspaces.

The following restrictions apply to the LIST / DBSPACESET command:

- When ON-Archive is in OPERATOR privilege mode, only **informix** and **root** can list information about dbspace sets.
- When ON-Archive is in GROUP privilege mode, only users in the **super_archive** group can list information about dbspace sets.


LIST
/DBSPACESET =
*
dbspsetname
List/dbspaceset Qualifiers
List/dbspaceset Qualifiers
1
/REQUEST =
rid
(
rid
)
,
*
1
Before and Since Qualifiers
p. 8-75
/USER =
user
*
1
/VSET = vsetname
/VOLUME =
volnum
1
Output Qualifiers.
p. 8-77
1
/DBSPACE =
dbspace
,
(
dbspace
)
*

DBSPACESET	specifies the dbspace set or sets about which to list information.	
	*	specifies all dbspace sets.
	dbspsetname	the name of a specific dbspace set.
REQUEST	specifies archive requests made for each dbspace set.	
	rid	specifies a request id in the catalog.
	*	specifies all requests that the user created for the specified dbspace sets.
USER	specifies other users who created requests. Whether you can see requests created by other users depends on the privileges you possess and the restrictions that apply to this command.	
	user	is a valid operating-system user id.
	*	specifies all users.
VSET	lists only dbspace sets contained in a specified vset.	
	vsetname	is a valid vset name.
VOLUME	lists only dbspaces sets contained on the specified volume.	
	vol	is a valid volume number.
DBSPACE	specifies dbspace and blobspace names about which to list information.	
	dbspace	is a dbspace or blobspace in the dbspace set(s) specified in DBSPACESET.
	*	specifies all dbspaces in the dbspace set.

LIST/DBSPACESET Example

The following LIST/DBSPACESET command requests a full listing of information for all dbspace sets (DBSPACESET=*), though only one dbspace exists:

```
Onarchive> LIST/DBSPACESET=*/FULL

Dbspace set : *
Dbspace(s) : (rootdbs)

Request ID : 00000001 Archive date : 06-Dec-1993 Compression : NO
Nb. copies : 01 Username : informix Encryption : NO
Expires : 06-Dec-1993 Archive level : 00 Starting Log: 00000006

Dbspace nameVolume-set nameVolume
---------------------------------
rootdbsv30001


Total of 1 dbspace set(s)
```

Tip: *The Starting Log value enables you to identify the logs you no longer need in the event of a restore. For example, when you list all of your dbspace sets and the minimum Starting Log value is 13, you know that you do not need to keep logs 1 through 12 anymore. Furthermore, when logs 1 through 12 are located in save sets 00000010 and 00000011, you can remove or expire those save sets, unless they also contain later logs (that is, providing that log 13 is also in save set 00000011, for example).*

Working with Backed Up Logical-Log Files

ON-Archive provides the following commands for working with backed-up logical-log files:

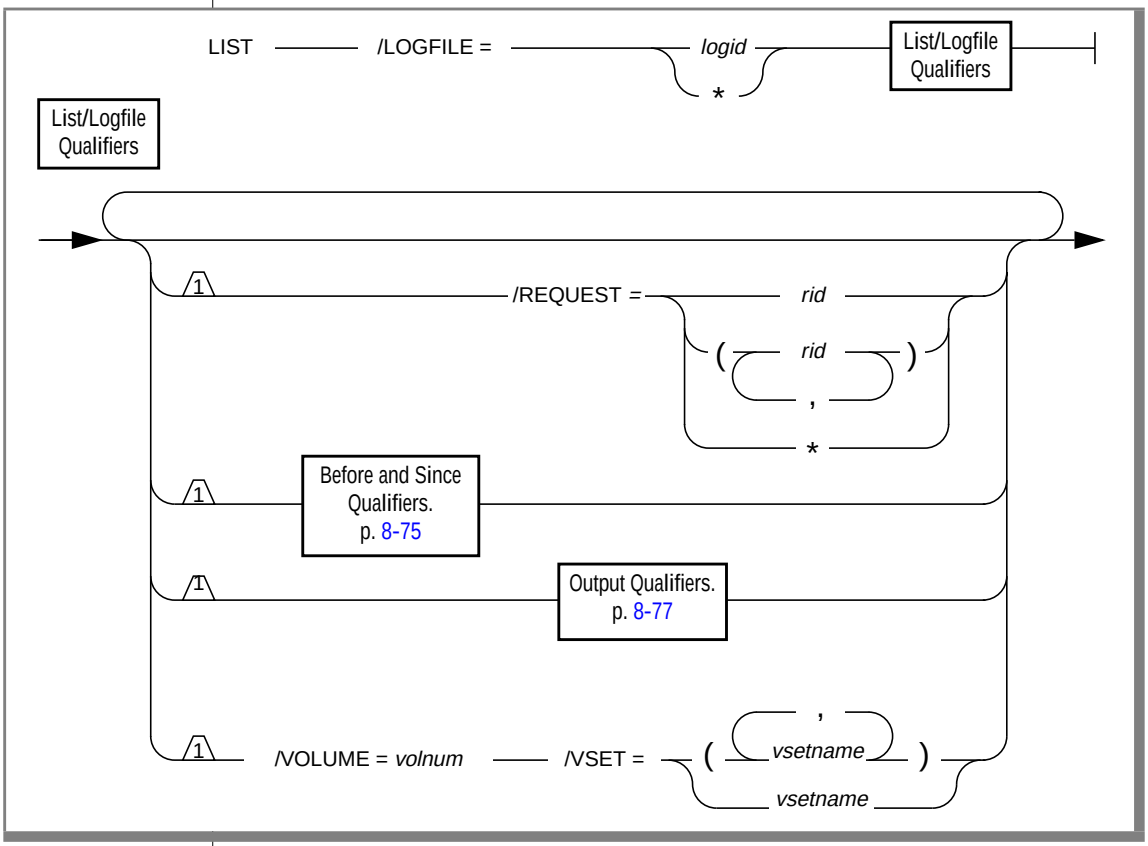
Command	Purpose
LIST/LOGFILE	Displays information about logical-log files that ON-Archive has backed up.
LIST/LOGRECORDS	Displays the contents of a logical-log file that ON-Archive has backed up.

The LIST/LOGFILE Command

The LIST/LOGFILE command displays information about backed up logical-log files.

The following restrictions apply to the LIST/LOGFILE command:

- When ON-Archive is in OPERATOR privilege mode, only **informix** and **root** can list information about log files.
- When ON-Archive is in GROUP privilege mode, only users in the **super_archive** group can list information about log files.



LOGFILE	specifies a log file.	
	logid	specifies a unique log id.
	*	specifies all log files.
REQUEST	specifies logical-log-backup requests for the specified log files.	
	rid	is a request id in the catalog.
	*	specifies all logical-log-backup requests.
VOLUME	lists all log files in a volume.	
	volnum	is a valid volume number in the specified vset.
VSET	lists all log files in a vset.	
	vsetname	is a valid vset name.

See "LIST/DBSPACESET Example" on page 8-55 for information on how to identify the log files you no longer need in the event of a restore.

LIST/LOGFILE Example

The following example lists log files, in both brief and full style:

```
Onarchive> LIST/LOGFILE=*/BRIEF
```

```
RID ArchiveUsernameCopiesFilename
-----------------------------------------
0000000519-May-1993usr01LF00000001
0000000519-May-1993usr01LF00000002
0000000519-May-1993usr01LF00000003
0000000519-May-1993usr01LF00000004
0000000519-May-1993usr01LF00000005
0000000519-May-1993usr01LF00000006
```

```
Onarchive> LIST/LOGFILE=2/FULL
```

```
File name: LF00000002
Request ID: 00000005Archive Date:19-May-1993Compression : NO
Nb. copies: 01Username: informixEncryption  : NO
Creation: 19-May-1993Modify date: 19-May-1993
Comment:

Volume-set nameVolume
----------------------
logset  0001
```

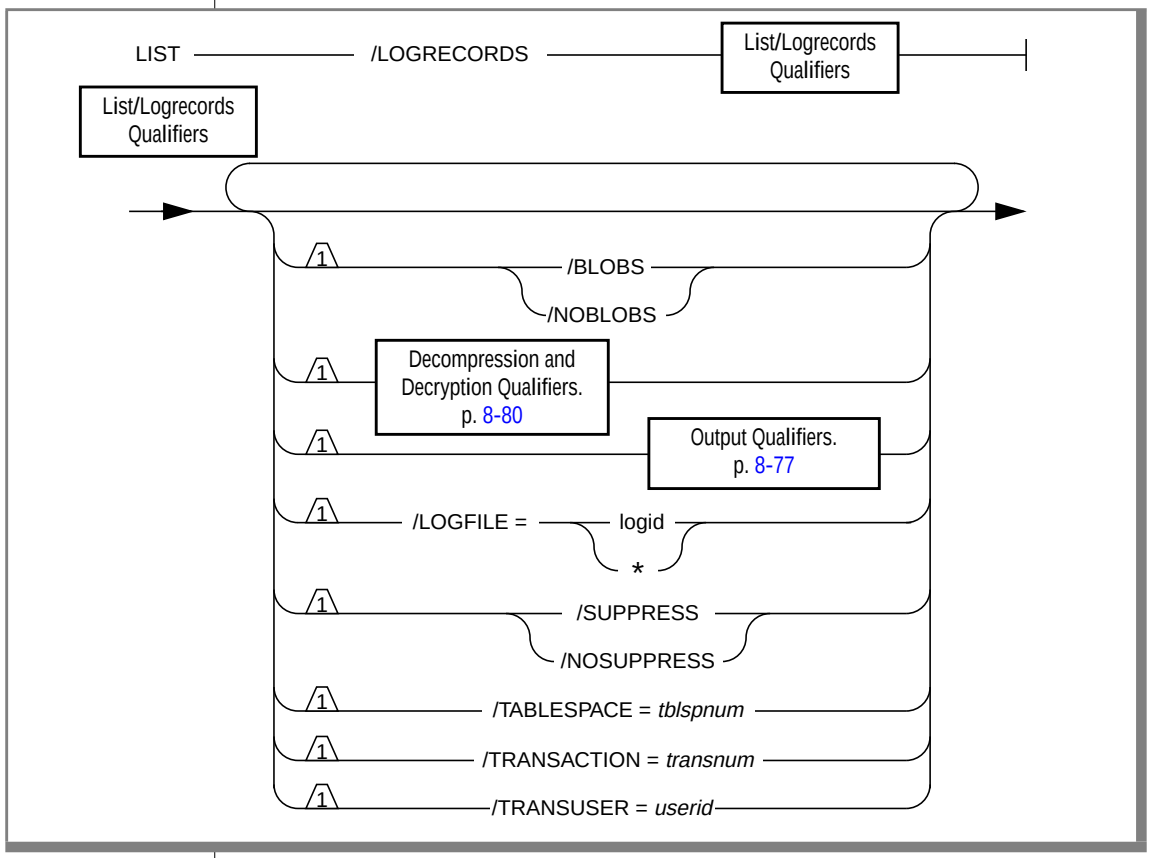
See "LIST/DBSPACESET Example" on page 8-55 for information on how to identify the log files you no longer need in the event of a restore.

The LIST/LOGRECORDS Command

The LIST/LOGRECORDS command displays the contents of a logical-log file backed up using ON-Archive. This uncommon operation usually only tracks transactions. See the *INFORMIX-Universal Server Administrator's Guide* for information on interpreting the output.

The following restrictions apply to the LIST/LOGRECORDS command:

- When ON-Archive is in OPERATOR privilege mode, only **informix** and **root** can list log records.
- When ON-Archive is in GROUP privilege mode, only users in the **super_archive** group can list log records.


LIST
/LOGRECORDS
List/Logrecords Qualifiers
List/Logrecords Qualifiers
1
/BLOBS
/NOBLOBS
1
Decompression and Decryption Qualifiers. p. 8-80
1
Output Qualifiers. p. 8-77
1
/LOGFILE =
logid
*
1
/SUPPRESS
/NOSUPPRESS
1
/TABLESPACE = tblspnum
1
/TRANSACTION = transnum
1
/TRANSUSER = userid

LOGRECORDS	lists the contents of a logical-log file that ON-Archive backed up.
BLOBS	displays blob data in the logical log.
NOBLOBS	suppresses blob data in the logical-log records. It is the default.
LOGFILE	displays a log file.
	logid specifies a unique log file. When no log file or * is specified, the command displays records in all the logical-log files.
SUPPRESS	suppresses a log header that occurs every 18 records.
NOSUPPRESS	does not suppress the log header. It is the default.
TABLESPACE	displays only records associated with the specified tblspace.
	tblspnum specified as either a decimal or hexadecimal value. (When you do not use an 0x prefix, you interpret the value as a decimal.)
TRANSACTION	displays only records associated with the specified transaction.
TRANSUSER	*transnum* a decimal or hexadecimal value.
	displays only records associated with activity that the specified user initiates.
	userid is a valid user id.

Press q to halt the log record output without quitting ON-Archive.

LIST/LOGRECORDS Example

The following example lists backed-up log records:

```
Onarchive> list/logrecords
```

```
log number: 10.

addr    len    type        xid  id   link
18      44     ADDITEM     4    1    f9790 1000e1 411 1 1 6
44      72     HINSERT     4    0    18 1000e1 412 36
8c      44     ADDITEM     4    0    44 1000e1 412 1 1 6
b8      72     HINSERT     4    0    8c 1000e1 413 36
100     44     ADDITEM     4    0    b8 1000e1 413 1 1 6
12c     72     HINSERT     4    0    100 1000e1 414 36
174     44     ADDITEM     4    0    12c 1000e1 414 1 1 6
1a0     72     HINSERT     4    0    174 1000e1 415 36
1e8     44     ADDITEM     4    0    1a0 1000e1 415 1 1 6
214     72     HINSERT     4    0    1e8 1000e1 416 36
25c     44     ADDITEM     4    0    214 1000e1 416 1 1 6
288     72     HINSERT     4    0    25c 1000e1 417 36
2d0     44     ADDITEM     4    0    288 1000e1 417 1 1 6
2fc     72     HINSERT     4    0    2d0 1000e1 418 36
344     44     ADDITEM     4    0    2fc 1000e1 418 1 1 6
370     72     HINSERT     4    0    344 1000e1 419 36
3b8     44     ADDITEM     4    0    370 1000e1 419 1 1 6
3e4     72     HINSERT     4    0    3b8 1000e1 41a 36

Press <RETURN> for more...
```

Changing Database Logging Status

ON-Archive provides the MODIFY / DBLOGGING command to enable you to change the logging status of a database.

The MODIFY/DBLOGGING Command

The MODIFY / DBLOGGING command marks the databases designated for changing their transaction-logging status. The following changes lock the database. The command considers the change pending. Thus, the change does not take place until after the next level-0 archive of all the dbspaces associated with the database.

From	To
no logging	unbuffered logging
no logging	buffered logging
no logging	ANSI-compliant

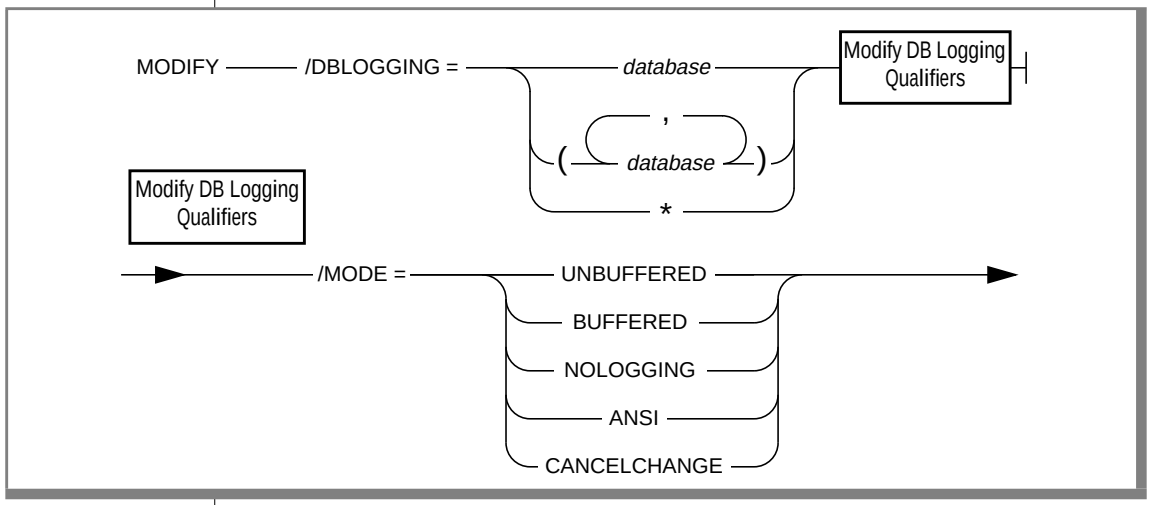
All other legal transitions (turning off logging, buffering changes, making a database with logging ANSI-compliant, and cancelling the request) take effect immediately.

You cannot change the logging status of a database from ANSI-compliant to no logging or buffered logging.

See the *INFORMIX-Universal Server Administrator's Guide* for more information on the logging modes that you can set using the MODIFY / DBLOGGING command.

The following restrictions apply to the MODIFY / DBLOGGING command:

- When ON-Archive is running in OPERATOR privilege mode, the user must log in as **informix** or **root** to modify the logging status of a database using ON-Archive.
- When ON-Archive is running in GROUP privilege mode, only members of the **super_archive** group can modify the logging status of a database using ON-Archive.



DBLOGGING changes the logging mode for the following database(s).

database	is a valid database that Universal Server manages.
*	specifies all databases.

MODE changes the logging mode of the database or databases to one of the following modes:

UNBUFFERED	changes the logging mode to unbuffered.
BUFFERED	changes the logging mode to buffered.

NOLOGGING	changes the database so it no longer uses transaction logging.
ANSI	changes the database to an ANSI-compliant database.
CANCELCHANGE	cancels a request to change the logging mode that requires a level-0 archive to take effect. This option takes effect after you execute a request, but before the level-0 archive that can activate the logging change.

Working with the Catalog

This section explains how to repair problems in the ON-Archive catalog or rebuild it when necessary.

Repairing Catalog Inconsistencies

The **arc_purge.sql** script validates the semantic constraints within the catalog and corrects any inconsistencies, such as leftover rows, that it encounters. Run the **arc_purge.sql** script periodically to remove any extraneous records that you possibly added to the catalog inadvertently.

Warning: *Make sure no other users are using ON-Archive when you run* ***arc_purge.sql*** *because it locks the ON-Archive catalog tables.*

The **arc_purge.sql** script is a script of SQL statements. You must use DB-Access to execute it, as illustrated in the following example:

```
% dbaccess - $INFORMIXDIR/etc/arc_purge.sql
```

See the *DB-Access User Manual* for information on how to use DB-Access.

The script produces messages that indicate the number of rows deleted from each of the tables in the catalog.

See "The arc_purge.sql Script" on page 9-4 for more information on **arc_purge.sql**.

Checking for Mismatches Between Volumes and the Catalog

When a request uses a disk vset, ON-Archive automatically checks for the following conditions:

- Whether the save sets in the catalog exist in the disk volumes
- Whether all the save sets found in the disk volumes exist in the catalog

When ON-Archive finds any inconsistencies, it creates an exception report and mails it to the user who executed the request.

When a save set exists in a disk volume but not in the ON-Archive catalog, the operator has the option of either deleting the save sets (using the operating system **rm** command) or adding them to the catalog using the ON-Archive CATALOG command. See "The CATALOG Command" on page 8-21 for more information.

When you register a save set in the ON-Archive catalog but it does not reside in the disk volume, the operator must remove it from the catalog using the ON-Archive REMOVE/REQUEST command. See "The REMOVE/REQUEST Command" on page 8-17 for more information.

Rebuilding the Catalog

When necessary, you can use the CATALOG command to rebuild the catalog. See "The CATALOG Command" on page 8-21.

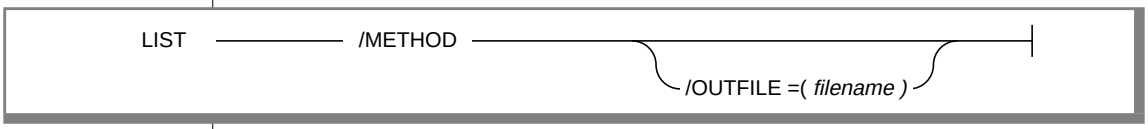
Listing Compression and Encryption Methods

ON-Archive provides the LIST / METHOD command to display compression and encryption information.

The LIST/METHOD Command

The LIST / METHOD command displays names and descriptions of the compression and encryption methods available to ON-Archive.

The LIST / METHOD command has no restrictions.



METHOD lists the names and descriptions of the compression and encryption methods available to ON-Archive.

OUTFILE specifies that you must send the output to a disk file instead of the screen. You cannot use the OUTFILE qualifier from the menu interface.

filename specifies the pathname and filename of the output file. You can use an absolute pathname, or a pathname relative to the directory where **onarchive** (or **onautovop**) was started. An absolute pathname must begin with a slash (/) and a relative pathname must *not* begin with a slash (/).

LIST/METHOD Example

```
Onarchive> LIST/METHOD

Encryption method list.
NAME         COMMENT

XOR          Bit eXclusive OR encryption.
             Very fast, but less secure.
             Parameter should have at least 3 characters.

Compression method list
NAME         COMMENT

REP          Standard compression method.
```

Groups of Qualifiers

The following sections describe groups of qualifiers that were referenced in the syntax diagrams for some commands. The LIST/DEFAULT command displays the default settings for ON-Archive qualifiers.

The following list shows these qualifier groups:

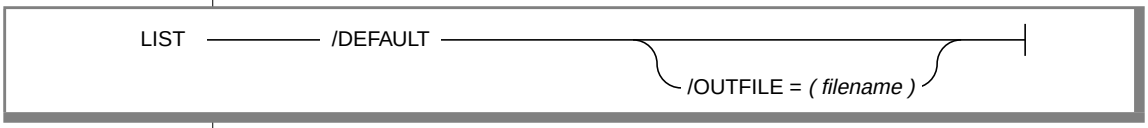
- ARCHIVE and BACKUP qualifiers
- BEFORE and SINCE qualifiers
- BRIEF, FULL, and OUTFILE qualifiers
- COMPRESS and ENCRYPT qualifiers
- DECOMPRESS and DECRYPT qualifiers
- PROTECTION qualifier
- WAIT and REPEAT qualifiers

The sections following the description of the LIST/DEFAULT command describe these groups of qualifiers. See "Using ON-Archive Commands" on page 4-7 for more information on qualifiers and their default values.

The LIST/DEFAULT Command

The LIST / DEFAULT command displays all current default settings for the ON-Archive qualifiers.

The LIST / DEFAULT command has no restrictions.



DEFAULT lists the current default settings for the ON-Archive qualifiers.

OUTFILE specifies that you must send the output to a disk file instead of the screen. You cannot use the OUTFILE qualifier from the menu interface.

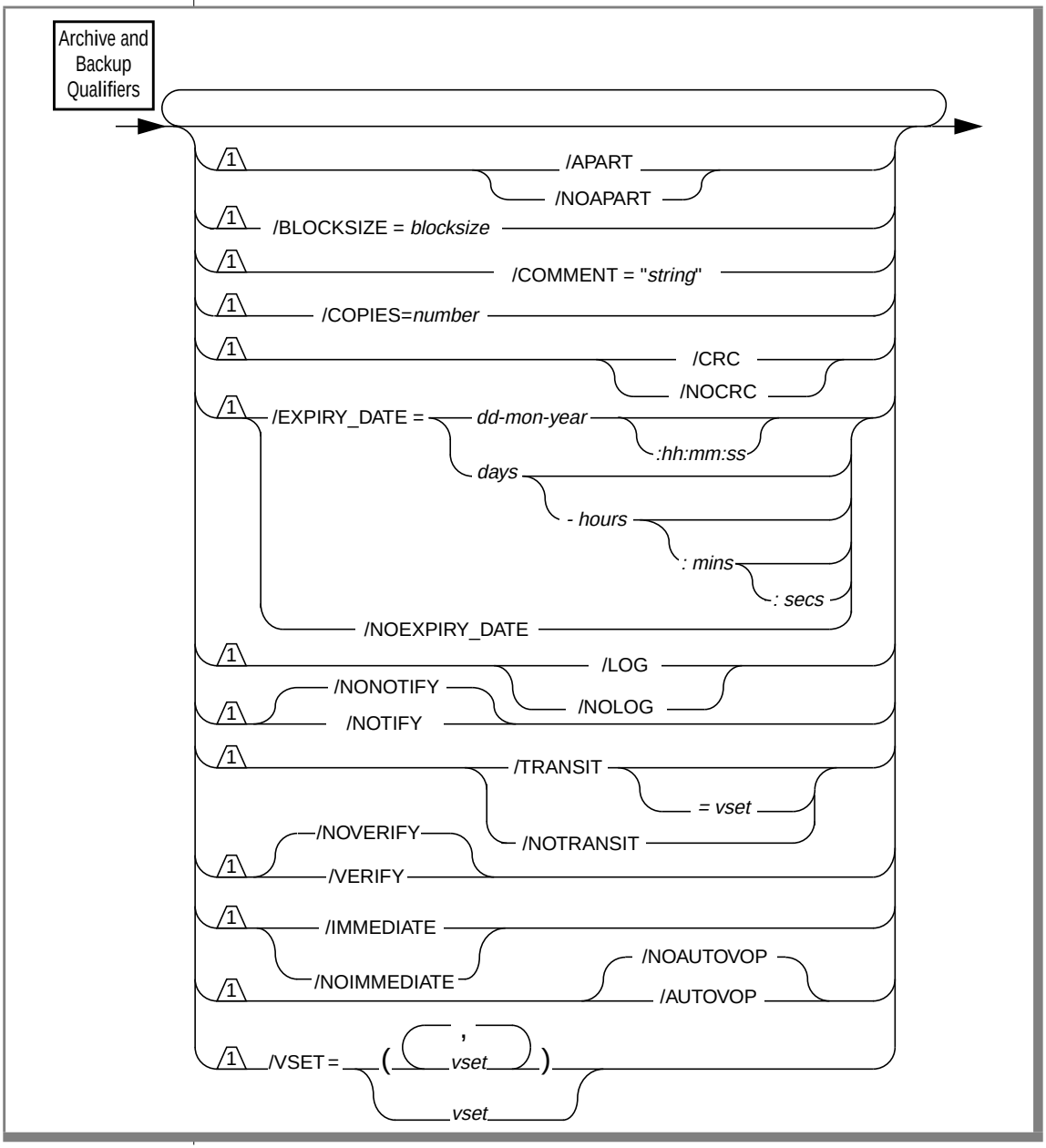
filename specifies the pathname and filename of the output file. You can use an absolute pathname, or a pathname relative to the directory where **onarchive** (or **onautovop**) was started. An absolute pathname must begin with a slash (/) and a relative pathname must *not* begin with a slash (/).

LIST/DEFAULT Example

The following example lists the default qualifier values for an ON-Archive installation on a computer called **peas**:

```
Onarchive> LIST/DEFAULT

SYSTEM PARAMETERS
-----------------

NOAPART              BLOCKSIZE=65024
BRIEF                NOCOMPRESS
COPIES=1             NODECOMPRESS
NODECRYPT            DENSITY=0
NOENCRYPT            NOEXPIRY_DATE
NOLOG                LEVEL=0
MAX_SPACE=0          NONOTIFY
OWNER_NODE=peas      PROTECTION=RWD
NOTRANSIT            USER=peas::informix
NOVERIFY
```

Archive and Backup Qualifiers



APART — specifies that the save set must reside on a separate volume from other save sets. That is, you write the save set to a blank volume, and you do not write any other save sets to the volume. Whenever you write to a remote tape device, you must use the APART qualifier. See "Keeping Archives on Separate Volumes" on page 5-17 of this manual.

NOAPART — indicates that the save set can reside on volumes with other save sets. It is the default.

AUTOVOP — specifies when you want to proceed to other tasks without waiting for a request to finish.

NOAUTOVOP — specifies the default setting for ARCHIVE and BACKUP qualifiers. Allows you to override an AUTOVOP default setting, specified in a personal default file, to ensure that the ARCHIVE and BACKUP qualifiers function as in earlier versions.

BLOCKSIZE — specifies the block size in bytes when writing to tape volumes or, with NB_DISK_SPACE_EXTENT, the size of disk-space allocations when writing to disk. See "Disk-Space Allocation During Concurrent Operations" on page 2-22 of this manual.

The internal default value is 64 kilobytes as specified in the operator default file. See "Where Qualifier Default Values Are Specified" on page 4-13 of this manual.

blocksize is an integer from 8197 to 65,024 specifying the block size in bytes.

COMMENT — specifies a text string containing a comment about the archive or backup operation. The comment resides with the request in the ON-Archive catalog.

string is a text string containing a comment about the archive. The string cannot exceed 80 characters.

COPIES

specifies the number of copies (including the original) that you must make of the save set that the request created. You create each copy on a different vset for greater security. See "Creating Multiple Copies of an Archive" on page 5-17 of this manual.

number is an integer from 1 to 5. You must specify the same number of vsets with the VSET qualifier.

CRC

adds a cyclic-redundancy check (CRC) field at the end of each data block during the backup. This check ensures that the data on the tape remains valid at the time of the restore.

NOCRC

bypasses the operation of writing the CRC field at the end of each data block during the backup. It is the default.

EXPIRY_DATE

specifies an expiration date for a request. The **onautovop** utility removes all information regarding the request from the ON-Archive catalog when the operating system date equals (or is greater than) the expiration date. See "Specifying When Archive Data Is Obsolete" on page 5-18 of this manual.

You can also remove specific requests manually (and thus render save sets obsolete) using the REMOVE/REQUEST command shown on page 8-17 of this manual.

dd is an integer between 1 and 31 representing a specific day in a month.

mon is a three-letter abbreviation for a month.

year is an integer representing a year.

hh is an integer representing the hour.

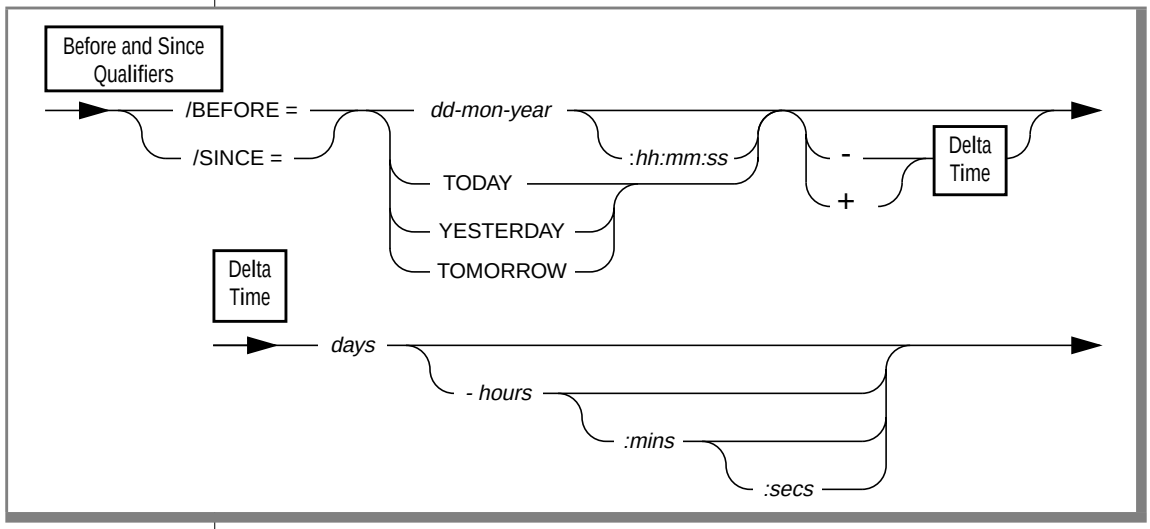
mm is an integer representing the minute.

ss is an integer representing the seconds in a minute.

	days	is an integer between 0 and 9999 specifying a number of days. It adds the value to the request-creation date (not the execution date) to determine the expiration date.
	hours	is an integer between 0 and 24 specifying a number of hours.
	mins	is an integer between 0 and 60 specifying a number of minutes.
	secs	is an integer between 0 and 59 specifying a number of seconds.
NOEXPIRY_DATE	specifies that you do not need to set an expiration date. It is the default.	
IMMEDIATE	simplifies the process of creating and executing a request. The IMMEDIATE qualifier saves time. It allows you to create and execute a request using a single command. It displays the request ID associated with that request entry.	
NOIMMEDIATE	specifies the default setting for ARCHIVE and BACKUP qualifiers. Allows you to override an IMMEDIATE default setting, specified in a personal default file, to ensure that the ARCHIVE and BACKUP qualifiers function as in earlier versions.	
LOG	specifies that you must generate a log file for this request. It creates the log file in the directory where the **onarchive** or **onautovop** command was started. The log file is named **arc***rid***.log**, where *rid* is the archive or backup request id.	
NOLOG	specifies that you cannot generate a log file for this request. It is the default.	
NOTIFY	specifies that after the request executes, the command sends electronic mail to the user who created the request.	
NONOTIFY	specifies that the command sends no mail after the request executes. It is the default.	

TRANSIT	specifies whether you can use a transit vset. See "What Is a Transit Volume Set?" on page 2-11 of this manual. When you use TRANSIT without any parameters, ON-Archive selects a transit vset. It searches first through the user's transit vsets for an available transit vset. When none is available, ON-Archive then searches the system-transit vsets. *vset* is the name of a transit vset.
NOTRANSIT	specifies that you cannot use a transit vset. It is the default.
VERIFY	instructs ON-Archive to reread each save set after writing it to tape. Rereading each save set ensures that you can later restore the data on the tape.
NOVERIFY	instructs ON-Archive not to reread each save set after writing it to the tape. NOVERIFY is the default.
VSET	specifies the vset(s) to use for this operation. No default value exists. When you do not specify a vset, the command selects one automatically. See "How ON-Archive Selects Volume Sets, Devices, and Volumes" on page 4-46 of this manual. The command usually specifies more than one vset only when it uses the COPIES qualifier. *vset* is the name of a vset.

Before and Since Qualifiers



BEFORE specifies that the operation is performed only against requests that were executed before the specified date.

SINCE specifies that the operation is performed only against requests that were executed after the specified date.

dd is an integer between 1 and 31 representing a specific day in a month.

mon is a three-letter abbreviation for a month.

year is an integer representing a year.

hh is an integer representing the hour.

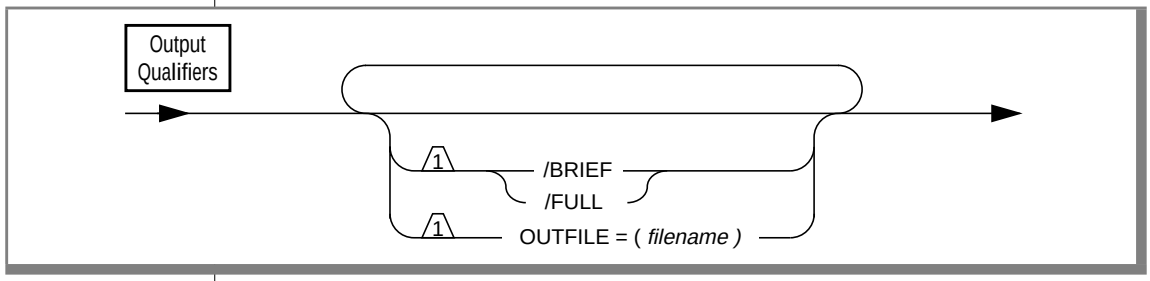
mm is an integer representing the minute.

ss is an integer representing the seconds in a minute.

TODAY is replaced with the current system date, month, and year at 00:00:00 o'clock.

YESTERDAY	is replaced with the system date, month, and year 24 hours before the value of the keyword TODAY.
TOMORROW	is replaced with the system date, month, and year 24 hours after the value of the keyword TODAY.
days	is an integer between 0 and 9999 specifying a number of days.
hours	is an integer between 0 and 24 specifying a number of hours.
mins	is an integer between 0 and 60 specifying a number of minutes.
secs	is an integer between 0 and 59 specifying a number of seconds.

Output Qualifiers



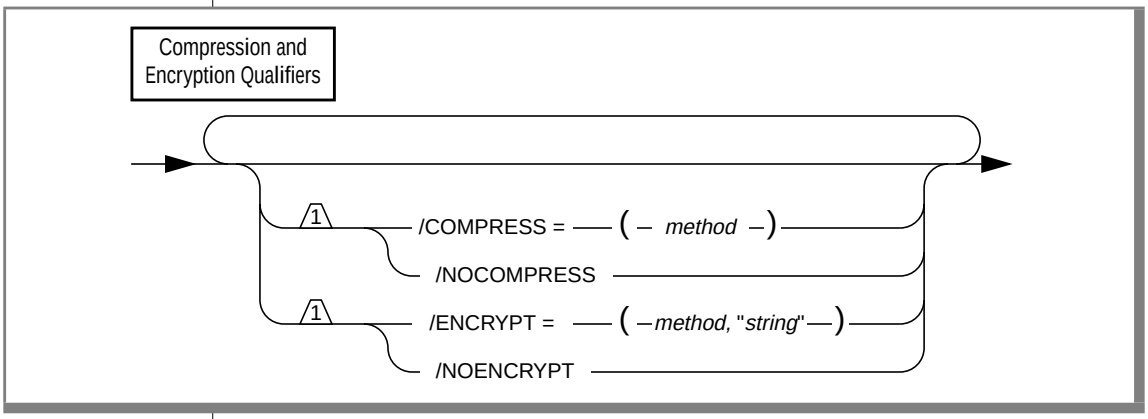
BRIEF displays only the basic information that the command produces.

FULL displays all the output the command produces.

OUTFILE specifies that you must send the output to a disk file instead of the screen. You cannot use the OUTFILE qualifier from the menu interface.

filename specifies the pathname and filename of the output file. You can use an absolute pathname, or a pathname relative to the directory where **onarchive** (or **onautovop**) was started. An absolute pathname must begin with a slash (/) and a relative pathname must *not* begin with a slash (/).

Compression and Encryption Qualifiers



COMPRESS indicates that you must compress the data before you write it to a volume. When you also require encryption, you perform encryption after you complete compression. For more information on compression, see "Data Compression" on page 2-24. When you compress the data, you must use the DECOMPRESS qualifier when you retrieve it.

method indicates the compression method. The local ON-Archive instance must know the compression method. Use the LIST / METHOD command on page 8-66 to get the list of the available compression methods.

NOCOMPRESS indicates that you must not compress the data before you write it to a volume. It is the default.

ENCRYPT indicates that you must encrypt the data before you write it to a volume. For more information on encryption, see "Data Encryption" on page 2-30. When you encrypt the data, you must use the DECRYPT qualifier when you retrieve it.

method indicates the encryption method. The local ON-Archive instance must know the encryption method.

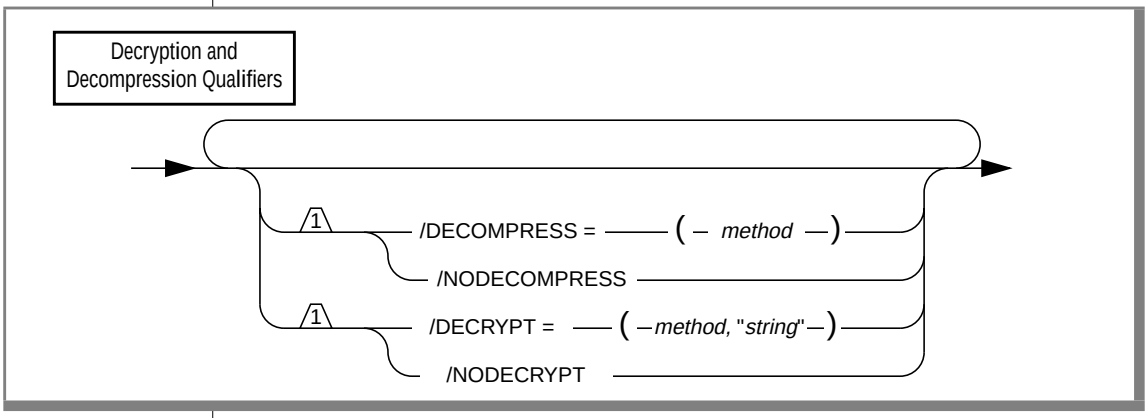
Use the LIST/METHOD command on page 8-66 to get the list of the available encryption methods.

string contains the arguments needed for the encryption routine (usually the encryption key). When the specified method requires more than one argument, you must include all arguments in the string.

NOENCRYPT indicates that data must not be encrypted before being written to a volume. It is the default.

You must contact Informix when you wish to use a compression or encryption method other than those that the LIST/METHOD command lists.

Decompression and Decryption Qualifiers



DECOMPRESS indicates that you must compress the data during the retrieve operation. You use it only when the COMPRESS qualifier was used when creating the archive or backup. When you require decryption, you must perform decryption before decompression.

method indicates the decompression method. The local ON-Archive instance must know the decompression method. Use the LIST / METHOD command on page 8-66 to get the list of the available decompression methods.

NODECOMPRESS specifies that you must not decompress the data.

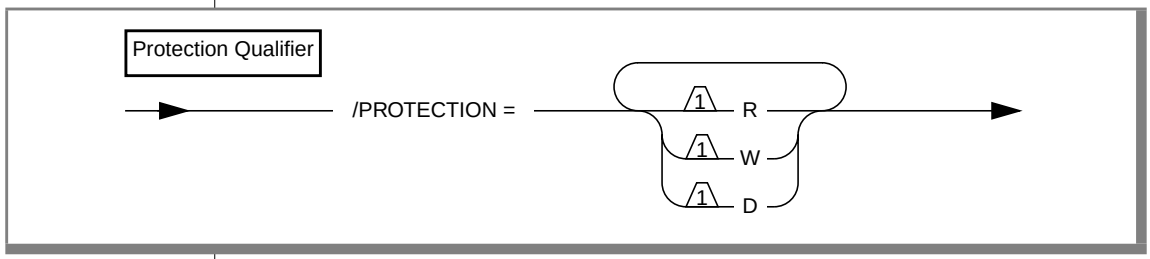
DECRYPT indicates that you must decrypt the data during the retrieve operation. You use it only after the ENCRYPT qualifier creates the archive or backup.

method indicates the decryption method. The local ON-Archive instance must know the decryption method. Use the LIST/METHOD command on page 8-66 to get the list of the available decryption methods.

string contains the arguments needed for the decryption routine (usually the decryption key). When the specified method requires more than one argument, you must include all arguments in the string.

NODECRYPT specifies that you must not decrypt the data.

Protection Qualifier



PROTECTION specifies the volume or vset protection. The following list shows the possible values:

R is read access.

W is write access. Write access is not allowed when you import the vset.

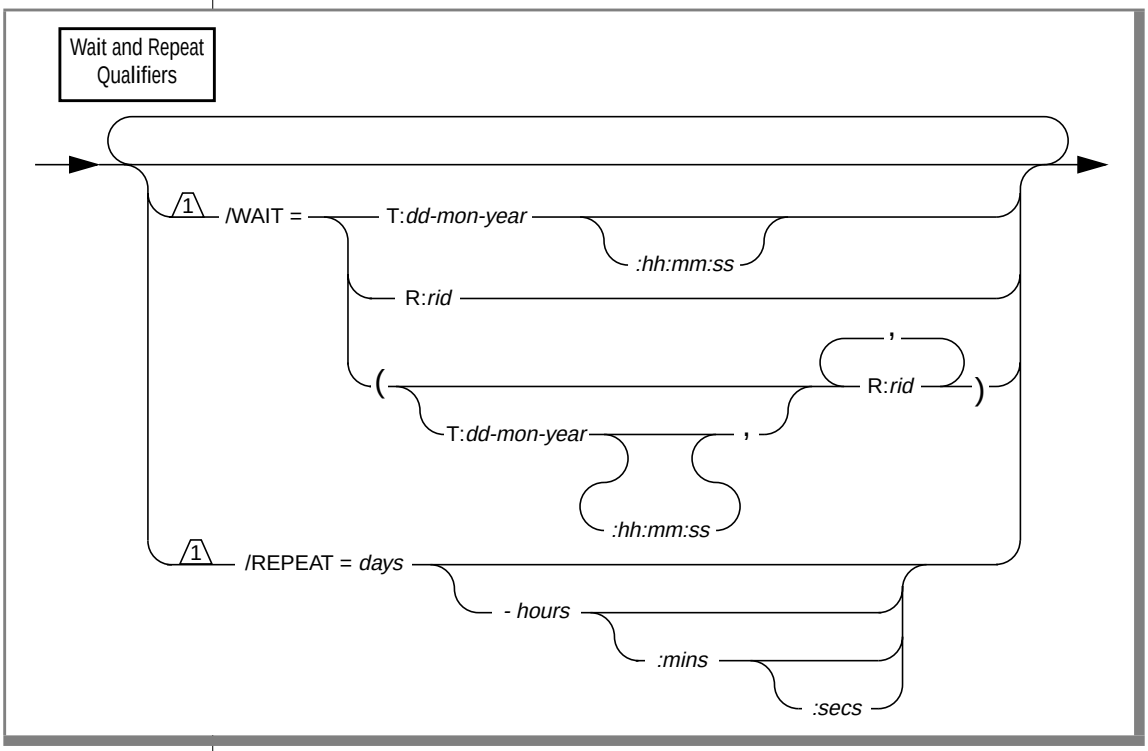
D is delete access.

The presence of R, W, and/or D in the string indicates that the particular access is enabled.

The default protection is RWD for vsets and volumes. See "Access Permission" on page 2-27 for more information on vset and volume protection.

When you grant a particular access to a vset, you do not automatically grant access to its volumes. To access the volumes, you must grant the same access to both the vset and the volume. You can change the volume protection and the vset protection at any time with the MODIFY / VOLUME command and the MODIFY / VSET command, respectively.

Wait and Repeat Qualifiers



WAIT specifies that the request must wait for the specified conditions to occur before you execute it (either manually by **onarchive** or automatically by **onautovop**). These conditions include a specified time, the completion of other requests, or both. When the wait depends on more than one other request, you cannot execute the current request until conditions finish occurring. When a condition request does not exist, or when it fails, the waiting request also fails.

T specifies a time.

dd is an integer representing the date.

mon is a three-character representation for a month.

year is an integer representing the year.

hh is an integer representing the hour.

mm is an integer representing the minute.

ss is an integer representing the seconds in a minute.

R specifies a request.

rid is a request id.

REPEAT specifies that you must submit a new request like the current one after the current one executes. The new request is the same as the original request, including the REPEAT qualifier, but with the addition of a new WAIT qualifier. To determine the value for the WAIT qualifier, add the delta, date, that the REPEAT qualifier specifies, to another date. You specify the delta date as shown in the following list:

days is an integer between 0 and 9999 specifying a number of days.

hours is an integer between 0 and 24 specifying a number of hours.

mins is an integer between 0 and 60 specifying a number of minutes.

secs is an integer between 0 and 60 specifying a number of seconds.

The other date is determined in the following way:

When the request does not use WAIT qualifier, the delta date is added to the date the request was executed.

When the request uses the WAIT=R qualifier, the delta date is added to the request execution time.

When the request uses the WAIT=T qualifier, the delta date is added to the request wait time.

When the request uses both the WAIT=T and WAIT=R qualifiers, the WAIT=R qualifier takes precedence.

Important: *The request must execute successfully for you to submit a new request. ON-Archive notifies you when the request fails.*

When you want to stop repeating requests, you can modify the request before it executes to remove the REPEAT qualifier. See "The MODIFY/VOLUME Command" on page 8-42.

Chapter

9

ON-Archive Utilities and Scripts

This chapter describes the syntax and usage of utilities and scripts for ON-Archive. For general information about ON-Archive, see Chapter 2, "What Is ON-Archive?"

To use all the scripts and utilities described in this chapter, you must set the following environment variables appropriately:

- **INFORMIXDIR**
- **INFORMIXSERVER**
- **ONCONFIG**

You must also set the **ARC_CONFIG** environment variable to the name of your ON-Archive configuration file when you do not use the **config.arc** file. See the *Informix Guide to SQL: Reference* for descriptions of these environment variables.

The arc_purge.sql Script

The **arc_purge.sql** script removes inconsistencies from the ON-Archive catalog. It is a script of SQL statements.

You need to log in as **informix** to successfully run **arc_purge.sql**.

Because it is an SQL script, you must use DB-Access to run **arc_purge.sql** from the operating-system command line, as shown in the following example:

```
% dbaccess - $INFORMIXDIR/etc/arc_purge.sql
```

The *DB-Access User Manual* describes how to use DB-Access.

The cron_autovop Utility

The **cron_autovop** utility provides an easy way to start **onautovop** automatically at specified times. It modifies your **crontab** file, allowing you to take the following actions:

- Add an entry to specify when you want **onautovop** to start
- Remove existing entries

You do not need to run the **oncatalgr** utility to use **cron_autovop**.

Any user can run **cron_autovop**. The **onautovop** process that **cron** subsequently starts runs as the user who created the **crontab** entry using **cron_autovop**. For more information on the **crontab** file, see your UNIX operating-system documentation.

cron_autovop

The **cron_autovop** utility prompts you to add or remove **crontab** entries. When you select to add an entry, it prompts you for the following information:

- List of hours (0 - 23)
- List of minutes (0 - 59)
- List of days of the month (1 - 31)
- List of months (1 - 12)
- List of days of the week (0:sunday - 6: saturday)

The values you enter specify when **onautovop** starts automatically. Enter an asterisk (*) to specify all the values for a given field and use commas to separate multiple entries. Press RETURN to leave a field empty. Do not use spaces to indicate an empty field.

The **cron_autovop** utility then prompts you for how many **onautovop** processes to start at that time.

Examples of cron_autovop Entries

For example, to start **onautovop** every day at 9:30 p.m., you can start the **cron_autovop** utility. It begins when it displays the following message:

```
odyssey% cron_autovop
```

```
The ONAUTOVOP utility is controlled by the Unix cron utility.
Do you want to add a new one? (y/n)
```

Enter y to add a new entry to the **crontab** file. The script prompts you, and you enter the time of day, the days of the month, the months, the days of the week and the number of **onautovop** processes that you want to start, as shown in the following example:

```
This script will modify your crontab file. It will add an
entry into it to define the time scheduling that specifies
when the ONAUTOVOP utility is to be started.

List of hours (0-23)                              : 21
List of minutes (0-59)                            : 30
List of days of the month (1-31)                  : *
List of months (1-12)                             : *
List of days of the week (0:sunday - 6:saturday) : *

How many onautovops should be started each time ? : 1
```

The **cron_autovop** script displays the following messages to confirm your entry in the **crontab** file:

```
Adding entry into crontab...


The ONAUTOVOP utility is now registered as a cron entry.
```

To start **onautovop** Mondays, Wednesdays, and Fridays at 8 o'clock in the evening, and on Saturday and Sunday at 10 o'clock in the morning, you use **cron_autovop** twice, entering the following values the first time it prompts you:

```
List of hours (0-23)                                : 20
List of minutes (0-59)                              : <RETURN>
List of days of the month (1-31)                    : <RETURN>
List of months (1-12)                               : <RETURN>
List of days of the week (0:sunday - 6:saturday) : 1,3,5
```

The second time you run **cron_autovop**, you enter the following values when it prompts you:

```
List of hours (0-23)                                : 10
List of minutes (0-59)                              : <RETURN>
List of days of the month (1-31)                    : <RETURN>
List of months (1-12)                               : <RETURN>
List of days of the week (0:sunday - 6:saturday) : 0,6
```

The onarchive Utility

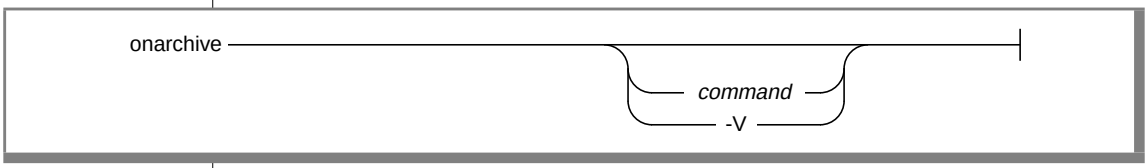
The **onarchive** utility lets you enter ON-Archive commands and execute ON-Archive requests. Chapter 4, "Using ON-Archive," lists all the ON-Archive commands and explains how to enter them.

As described in "Using the ON-Archive Command-Line Interface" on page 4-16 and "Using the ON-Archive Menu Interface" on page 4-17, you can submit commands to **onarchive** either from a command line or through a menu.

You must run the **oncatlgr** utility to use **onarchive**.

You can run multiple **onarchive** processes at the same time for an INFORMIX-OnLine Dynamic Server instance.

Any user can run **onarchive**, although you can encounter restrictions on the individual commands. See "Using ON-Archive Commands" on page 4-7 for a complete list of ON-Archive commands, including a cross reference to the detailed discussion for each command.



command is any ON-Archive command listed in "Using ON-Archive Commands" on page 4-7.

-V displays the version number of Universal Server and then quits.

When you provide no *command*, **onarchive** enters into an interactive mode. You can enter more commands at the following prompt:

```
Onarchive>
```

The **onarchive** utility returns 1 when it encounters an error.

The onautovop Utility

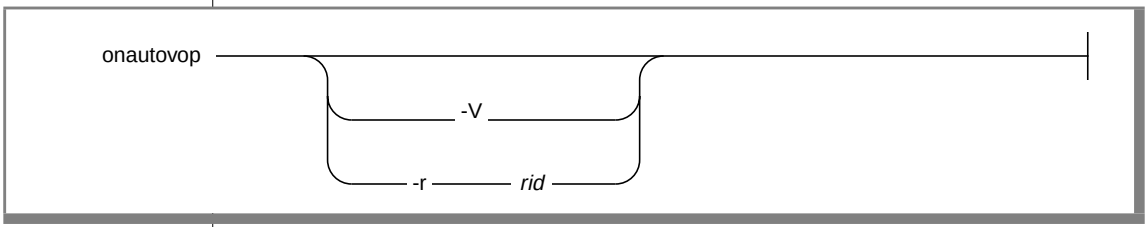
The **onautovop** utility enables you to execute requests automatically and without the need for an operator. "Using Unattended Operations" on page 4-50 describes the steps you must perform to execute a request automatically. It also describes how **onautovop** selects volume sets (vsets) and volumes for the requests that it executes. Before you start **onautovop**, you must define and mount all the volumes needed for any requests that **onautovop** processes.

See "The start_autovop Script" on page 9-31 for information on starting **onautovop**. Although **onautovop** stops once it has carried out all its tasks, you can also stop it manually. See "The stop_autovop Script" on page 9-33 for information on stopping **onautovop** manually.

You can put an entry into the UNIX **crontab** file to execute the **start_autovop** script at regular intervals. Or, when do not know about the **crontab** file, you can use the **cron_autovop** utility to help you create an entry in the **crontab** file. See "The cron_autovop Utility" on page 9-5 for information on **cron_autovop**.

Any user can run **onautovop**. When you run **onautovop** using **root** or **informix** (and ON-Archive is in OPERATOR privilege mode) or using a user in the **super_archive** group (and ON-Archive is running in GROUP privilege mode), **onautovop** executes all pending requests. When any other user runs **onautovop**, **onautovop** only executes the pending requests that user submitted

You can run multiple **onautovop** processes for an Universal Server instance at the same time. When you use the **onautovop** utility, the **-r** command-line qualifier allows you to execute a specific request. The **onautovop** utility executes only the specified request ID and then exits. The following diagram shows the syntax of the **onautovop** utility.



-r executes the request specified by the *rid* qualifier.

rid represents the request ID number.

-V prints the version number and exits.

The following example shows how to use the **-r** option with **onautovop**:

```
onautovop -r 5
```

In the example, the value 5 represents a request.

When an ON-Archive error occurs while **onautovop** is running, ON-Archive sends that error to the creator of the request that caused the error.

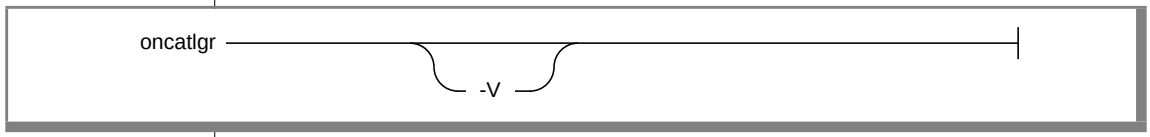
You can run the **onautovop** utility in the background or start it with **cron**. To specify times for **cron**, you can use the **cron_autovop** utility, described in "The cron_autovop Utility" on page 9-5.

The oncatlgr Utility

The **oncatlgr** utility manages information coming in and out of the ON-Archive catalog. You must run the **oncatlgr** utility to invoke **onarchive** and **onautovop**.

As described in "Starting the Cataloger Automatically" on page 4-6, you can also start **oncatlgr** using the **start_oncatlgr** script.

You must log in as **root** or **informix** to run **oncatlgr**.



-V displays the version number of Universal Server and then quits.

When an **oncatlgr** error occurs, **oncatlgr** sends the error to standard output. When you start **oncatlgr** in the background (using the & at your operating-system prompt), you can redirect the output. You redirect the output when you run multiple Universal Server instances on the same computer, and you want to use the **-f** option of **stop_oncatlgr** to stop **oncatlgr**. See "The stop_oncatlgr Script" on page 9-34.

When you start **oncatlgr** using the **start_oncatlgr** script, **oncatlgr** sends output to a file in the directory **/tmp**. See "The start_oncatlgr Script" on page 9-32 for information on the name of the file.

To fix most cataloger errors, simply stop and start **oncatlgr**.

Utility Enhancements That Automatically Start oncatlgr

Running the **onarchive** or **onautovop** utilities automatically starts the **oncatlgr** process.

Both **onarchive** or **onautovop** require a running **oncatlgr** process to operate. When either of those utilities start, they check for a running **oncatlgr** process. When that process is not running, either utility automatically starts an **oncatlgr** process.

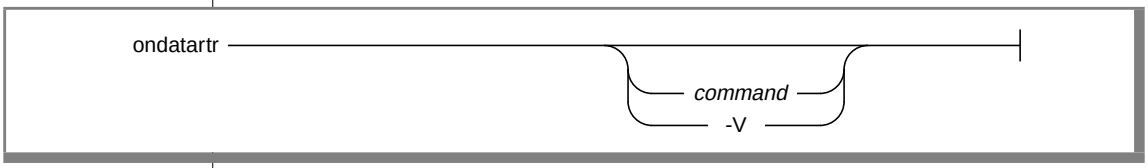
The ondatartr Utility

As described in "Emergency Situations" on page 4-48, the **ondatartr** utility lists and retrieves data from an ON-Archive save set without using the ON-Archive catalog or the **onarchive** utility. This feature allows you to access your archived and backed-up data on tape or disk even when your Universal Server database server or **oncatlgr** is not running. You must use the **ondatartr** utility in the following situations:

- When you need to perform a cold restore (see "A Cold Restore" on page 7-7)
- When you need to perform an emergency logical-log-file backup (see "Performing Emergency Logical-Log File Backups" on page 6-13)

You must log in as **informix** or **root** to use **ondatartr**.

Universal Server must shift to off-line mode for **ondatartr** to perform a cold restore and in on-line or quiescent mode for **ondatartr** to perform an emergency log-file backup.



command is one of the commands listed in "The ondatartr Command-Language Syntax" on page 9-14.

-V displays the version number of Universal Server and then quits.

When you do not supply a *command* or **-V** supplied, **ondatartr** enters into an interactive mode where you can enter commands, as shown in the following example:

```
% ondatartr
ONDATARTR>
```

The prompt indicates that the **ondatartr** utility can now accept a command.

You enter commands to the right of the prompt and that terminate when you press the RETURN key. The **ondatartr** utility processes the command and displays a new prompt when ready for another command.

You can enter a command on several lines. When a command exceeds the length of one line, end the command with a dash (-). A new prompt _ONDATARTR> starting with an underscore, indicates that **ondatartr** expects the continuation of the command line.

Getting Help with ondatartr

You can get help when using **ondatartr** commands. To get help on a specific topic, enter the command HELP and type the desired topic, as shown in the following example.

```
ONDATARTR> HELP RETRIEVE
```

```
RETRIEVE
Help text on RETRIEVE command ...
Additional information available:
List of sub-topics ...
                        RETRIEVE sub-topic?
To get help on any of the RETRIEVE sub-topics, enter the sub-
topic name at the "sub-topic" prompt.
To quit help, press [RETURN] until the ondatartr command line
prompt appears.
```

Using More Than One ondatartr Program

You can start and use more than one **ondatartr** program at one time during a physical restore, with the following provisions:

- Each **ondatartr** program must read from a different save set during parallel restores.
- Universal Server must shift to recovery mode.

You do not need to run more than one **ondatartr** program for a logical restore. During a logical restore, Universal Server automatically rolls forward the logical-log files in parallel.

Exiting ondatartr

Use the command EXIT to quit **ondatartr**.

```
ONDATARTR> EXIT
%
```

Interrupt Enhancement to the ondatartr Utility

Interrupting a restore forces you to restart the entire process from the beginning. When you interrupt a logical restore by pressing CTRL-C, **ondatartr** now verifies whether you want to proceed with that interrupt. The verification prompt asks you if you want to interrupt **ondatartr** and provides you with a last chance to prevent the termination of a logical restore.

When you press the CTRL-C key sequence during a logical restore, the **ondatartr** utility displays the following prompt:

```
Interrupt of ondatartr during logical restore will require a
restart of the entire cold restore. Really interrupt the
restore? (y/n)
```

Type Y to terminate **ondatartr**. Type N to continue the logical restore without interruption.

The ondatartr Command-Language Syntax

The command language for **ondatartr** uses the same syntax as **onarchive**. Therefore, **ondatartr** obeys the same rules regarding command qualifiers, qualifier specifications, qualifier ordering, qualifier overriding, case, and abbreviations as explained in "Command and Qualifier Syntax" on page 4-10.

The **ondatartr** commands fall into several categories, as described in the following sections.

List Commands

These commands display all the information on dbspace sets or log files on a specific volume (tape or disk).

- "LIST / DISK Command" on page 9-20
- "LIST / TAPE Command" on page 9-22

Retrieve Commands

These commands allow the retrieval of archived data and logical-log files without connecting to your Universal Server database server. You can only use them when the database server remains off-line.

- "RETRIEVE / DBSPACESET Command" on page 9-24
- "RETRIEVE / LOGFILE Command" on page 9-27

  During a cold restore, the RETRIEVE / LOGFILE command allows you to use the UNTIL qualifier to restore data to a specific point in time.

Backup Log-File Command

This command allows you to back up a logical-log file when Universal Server has suspended processing because of a full logical log. See "BACKUP Command" on page 9-16.

Exit and Help Commands

These commands allow you to exit the **ondatartr** utility and display Help information, respectively.

- "EXIT Command" on page 9-18
- "HELP Command" on page 9-19

The following pages describe each **ondatartr** command and its command qualifiers.

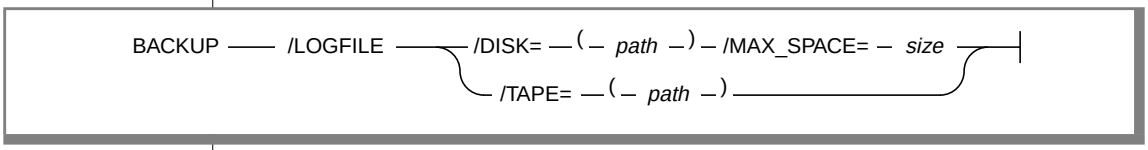
BACKUP Command

As described in "Performing Emergency Logical-Log File Backups" on page 6-13, the BACKUP command permits you to back up logical-log files in the event that Universal Server processing stops because of full logical-log files (or, more precisely, when the logical log reaches a percentage of fullness that the LTXEHWM parameter specifies in the ONCONFIG file).

The BACKUP command backs up log files to a vset called ONDATARTRLOG and starts with the volume number 1.

As described in "Performing Emergency Logical-Log File Backups" on page 6-13, after **ondatartr** creates the log-backup volumes, you can use the CATALOG command to catalog the volumes. You must then copy the cataloged volumes to the vset where you regularly store logical-log files, and then remove them from the ONDATARTRLOG vset.

Informix recommends that you back up log files using **onarchive** or **onautovop** rather than **ondatartr**. You use the **ondatartr** utility only when Universal Server processing stops because of a full logical log.



LOGFILE specifies that you must back up all full log files.

DISK specifies the pathname to the directory that becomes the first volume where logical-log files back ups reside.

path is the pathname to an operating system directory.

MAX_SPACE specifies the maximum amount of space (with an integer representing a number of 512-byte blocks) that **ondatartr** uses for this volume.

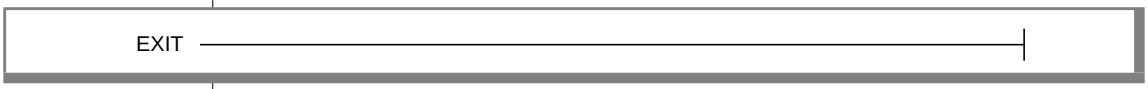
size is an integer between 0 and 99999999. You can specify a minimum usable MAX_SPACE size of 16 for a computer with a page size of 2 kilobytes, or 28 for a computer with a page size of 4 kilobytes.

TAPE specifies the pathname to the device where you mount a blank tape that becomes the volume that holds the backed up the logical-log files.

path is the pathname to a tape device.

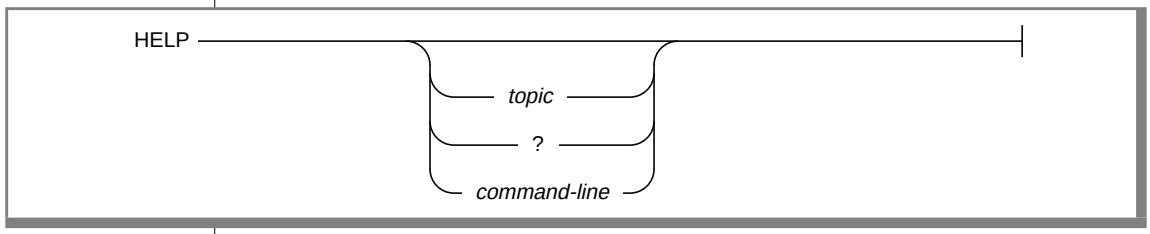
EXIT Command

The EXIT command exits the **ondatartr** utility.



HELP Command

The HELP command displays information about an **ondatartr** command or topic. See "Getting Help with ondatartr" on page 9-13.



topic is any topic for which you require information.

? is a literal question mark. It tells you how to use Help.

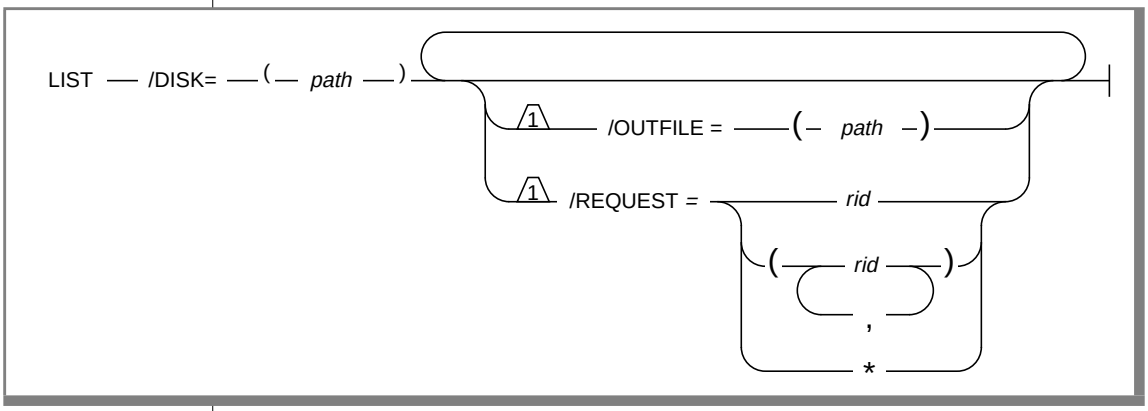
command-line is the name of any **ondatartr** command.

The following example causes your monitor to display the first Help screen of the LIST command:

```
ONDATARTR> HELP LIST
```

LIST/DISK Command

The LIST / DISK command displays information on dbspace sets and logical-log files stored on a disk volume. It also reads and verifies the cyclic redundancy check (CRC) on the save-set blocks, when you specified the CRC qualifier when the save set was created.



DISK specifies the directory where the disk volume was defined. The directory specification must not contain any wildcard characters.

OUTFILE specifies the output device that displays the list. When you do not specify a device, your terminal becomes the output device.

path is the pathname to a file.

REQUEST lists only those save sets on the volume that the specified archive requests create.

rid is a valid request id.

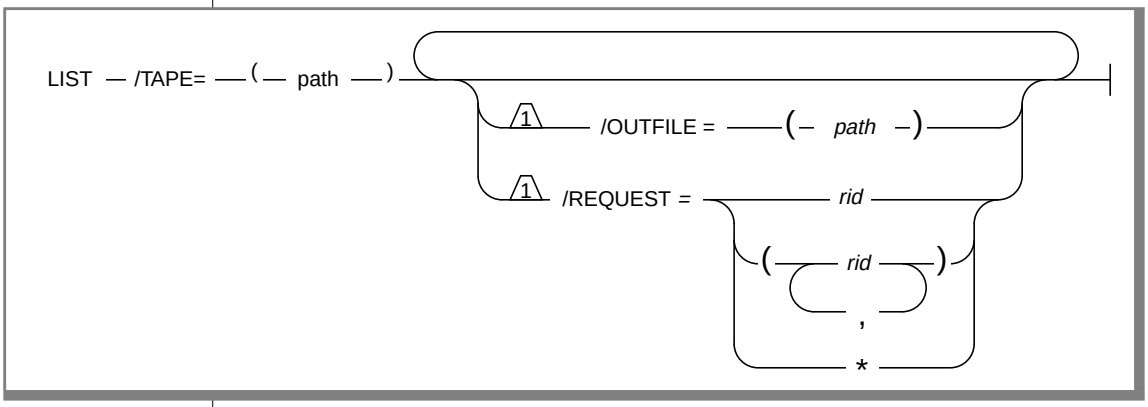
The following example lists information on a disk volume:

```
ONDATARTR> LIST/DISK=(/usr/accvset/vol1)/REQUEST=162
```

```
LISTING OF : accvset
VOLUME #    : 0001
--------------------
ON-ARCHIVE Version  : ON-Archive VERSION 6.0
Save-set file name  : /usr/accounting/vol1/00000162.SAV
Archive date        : 20-AUG-1993 11:44:29
Save-set type       : OS files
Operating system    : PAGE4096 IEEEM BYTE32
DBMS                : IXARCHDB_VER_06.00 LEVEL0
Owner node          : sparc
Owner name          : fletch
Archive Command     : ARCHIVE/DBSPACESET=accvset
                    : /LEVEL=0/COPIES=1/NOCOMPRESS/NOAPART
                    : /BLOCKSIZE=65024/NOENCRYPT/NOLOG
                    : /NOTIFY/NOVERIFY/NOTRANSIT/NOEXPIRY_DATE

File name           : rootdbs
File number         : 1
File sequence       : 1

File name           : dbs1
File number         : 2
File sequence       : 1
```

LIST/TAPE Command

The LIST/TAPE command displays information on dbspace sets and logical-log files written to a tape volume.



TAPE specifies the device name where you mount the listed volume.

path is the pathname to a tape device.

OUTFILE specifies the output device that displays the list. When you do not specify one, your terminal becomes the output device.

path is the pathname to a file.

REQUEST lists only those save sets on the volume that the specified archive requests create.

rid is a valid request id.

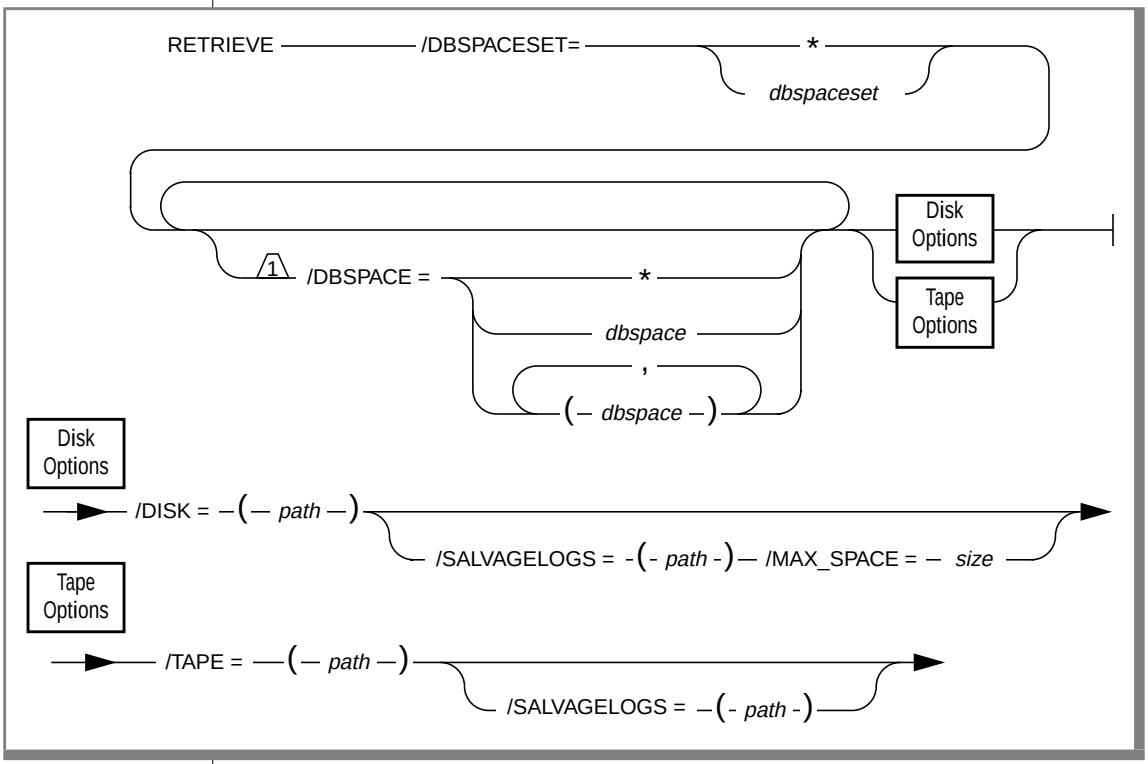
The following example lists information on a tape volume:

```
ONDATARTR> LIST/TAPE=(/dev/rst0)/REQUEST=162
```

```
LISTING OF : accvset
VOLUME #    : 0001
--------------------
ON-ARCHIVE Version  : ON-Archive VERSION 6.0
Save-set file name  : 00000162.SAV
Archive date        : 20-AUG-1993 11:44:29
Save-set type       : OS files
Operating system    : PAGE4096 IEEEM BYTE32
DBMS                : IXARCHDB_VER_06.00 LEVEL0
Owner node          : sparc
Owner name          : fletch
Archive Command     : ARCHIVE/DBSPACESET=accvset
                    : /LEVEL=0/COPIES=1/NOCOMPRESS/NOAPART
                    : /BLOCKSIZE=65024/NOENCRYPT/NOLOG
                    : /NOTIFY/NOVERIFY/NOTRANSIT/NOEXPIRY_DATE

File name           : rootdbs
File number         : 1
File sequence       : 1

File name           : dbs1
File number         : 2
File sequence       : 1
```

RETRIEVE/DBSPACESET Command

The RETRIEVE / DBSPACESET command performs a physical restore of the dbspace set or dbspaces specified. You must restore the root dbspace first before restoring other dbspaces. You can only use this command when Universal Server is off-line.



DBSPACE — specifies particular dbspaces that you restore within a dbspace set. When you omit this qualifier, you restore all the dbspaces within the dbspace set.

- *dbspace* — is any dbspace that was archived with this dbspace set.
- * — specifies all dbspaces for the dbspace set.

DBSPACESET — specifies the dbspace set or sets you intend to restore. Restore the root dbspace before restoring other dbspaces.

When you specify *, it refers to a restore of all the dbspaces Universal Server manages. To use * for a restore, you must, in advance, create the archive that you want to restore using the * parameter with the ARCHIVE / DBSPACESET command.

- *dbspaceset* — is any dbspace set that was archived in this save set.

DISK — specifies the pathname of the disk volume from which you restore the dbspaceset.

- *path* — is the pathname to a file.

MAX_SPACE — specifies the size of the disk volume in 512-byte blocks where you write the salvaged logs. You must specify MAX_SPACE when you specify the DISK qualifier.

- *size* — is an integer between 0 and 99999999. You specify 16 as the minimum usable size of MAX_SPACE, that equals ((page size * 3) / 512) + 4.

SALVAGELOGS instructs **ondatartr** to back up any logical-log files that you did not yet back up. It backs them up to the specified volume before a level-0 physical restore occurs. You can assume the device type (DISK or TAPE) as the same specified for the retrieve operation, although the pathname to the volume device can differ.

The volume created resides in the vset ONDATARTRLOG and uses volume number 1. You can later use the CATALOG command to catalog the volume.

path is the pathname to either a file or a tape device. Pathnames used here can differ from the pathnames specified in the DISK and TAPE qualifiers, and must be of the same type.

TAPE specifies the pathname of the device that has mounted on it the tape volume from which you can restore the dbspaceset.

path is the pathname to a tape device.

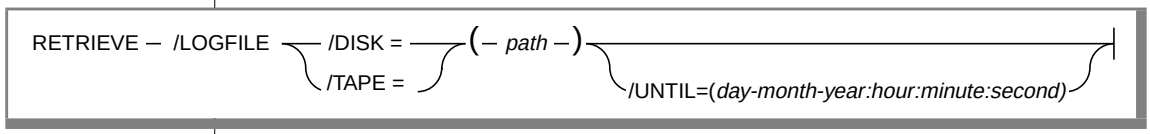
As described in "Start Parallel Cold Physical Restores" on page 7-28, once Universal Server shifts to recovery mode, you can use a different **ondatartr** process to start a physical restore of another save set (or sets).

As described in "Perform a Logical Restore with ondatartr" on page 7-30, once you complete the physical restore, you can use **ondatartr** to start a logical restore, or you can bring Universal Server to on-line mode without restoring the logical-log files. When you do not restore logical-log files, you restore Universal Server data only to its state at the time of the archive.

RETRIEVE/LOGFILE Command

The RETRIEVE / LOGFILE command determines those logical-log files that were backed up after the last archive, tells you the numbers, and prompts you to enter the request IDs of commands to retrieve them.

When Universal Server runs (in recovery mode) to execute this command, then the **oncatalgr** utility cannot run at the same time. After you retrieve the log files, Universal Server shifts to quiescent mode.



DISK specifies the pathname of the disk volume from which you restore the log files.

path is the pathname to a directory.

LOGFILE specifies that you start a logical-restore operation.

TAPE specifies the pathname of the device that has mounted on it the tape volume from which you restore the log files.

path is the pathname to a tape device.

UNTIL specifies the date and time to which you intend to restore data.

day is the two-digit day specified.

month is the month specified using three uppercase letters.

year is the four-digit year.

hour is the two-digit hour.

minute is the two-digit minute.

second is the two-digit second.

The UNTIL qualifier allows point-in-time recovery for data. Point in time recovery can only occur after you complete a physical restore (cold restore) and takes effect during a logical restore. When you do not specify the UNTIL qualifier, the logical restore functions as implemented at the saveset level.

The UNTIL qualifier performs a logical restore that recovers a transaction providing that the transaction began committing before the time specified in the UNTIL qualifier. You cannot recover data under the following conditions:

- When a transaction starts on or before the time that the UNTIL qualifier specifies, but does not commit until after that specified time
- When a transaction starts after the time that the UNTIL qualifier specifies

When you perform point-in-time recovery, the UNTIL qualifier does not mark the log file that contains the specified point in time as backed up. You must back up that log file again because the transaction history changed starting with that log.

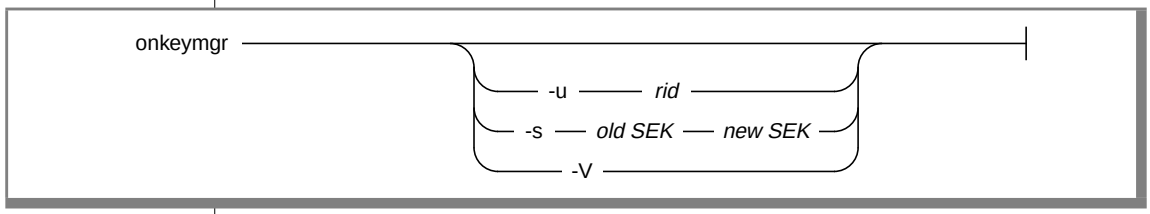
The following example restores the data to its state on August 24th, 1995 at 3:38 p.m. and 39 seconds:

```
ondatartr> retrieve/logfile/tape=(/dev/rmt/0m)/until=(24-AUG-1995:15:38:39)
```

The onkeymgr Utility

You use the key manager utility, **onkeymgr**, to change the system-encryption key (SEK) or to decrypt a forgotten user- encryption and method key (UEMK). For a discussion of encryption, see "Data Encryption" on page 2-30.

You must log in as **root** or **informix** to run **onkeymgr**.



-u returns the decrypted UEMK for the request specified by the request id.

rid is the request id that identifies the encrypted save set.

-s automatically reencrypts all the stored UEMKs in the catalog and restarts the **oncatalgr** process with the new SEK.

old SEK is the old value of the system-encryption key.

new SEK is a new value for the system-encryption key. This value must span between one and eight characters in length.

-V displays the version number of *Universal Server* and then quits.

When you run **onkeymgr** without options, a menu interface appears and presents you with the choice of either changing the SEK or recovering a lost UEMK.

When you use the menu interface to change the SEK, it prompts you for the following values:

- The current value of the SEK
- The new value of the SEK

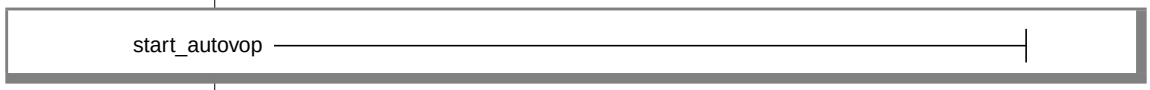
ON-Archive automatically reencrypts all the stored UEMKs in the catalog and restarts the **oncatalgr** process with the new SEK.

When you use the menu interface to decrypt a forgotten UEMK, ON-Archive prompts you for the request id that identifies the encrypted save set for which the UEMK is lost.

The start_autovop Script

The **start_autovop** script lets you start an **onautovop** process.

Any user can run **start_autovop**.



When you start **onautovop** with **start_autovop**, errors that **onautovop** generates get directed to **/tmp/onautovop.out.*pid***, where *pid* is the process identification number of the **start_autovop** process.

To fix most errors, simply stop and restart **onautovop**.

The start_oncatlgr Script

The **start_oncatlgr** script lets you start an **oncatlgr** process. When you already have an **oncatlgr** process (or several) running, it prompts you with the option of killing each existing **oncatlgr** process before you start the new one. When you use more than one Universal Server instance on a single computer, you could encounter multiple running **oncatlgr** processes. Thus, take care to stop only the **oncatlgr** process associated with the correct Universal Server instance. See "Which oncatlgr for Which Universal Server?" on page 4-6 for more information.

You must log in as **root** or **informix** to run **start_oncatlgr**.

start_oncatlgr

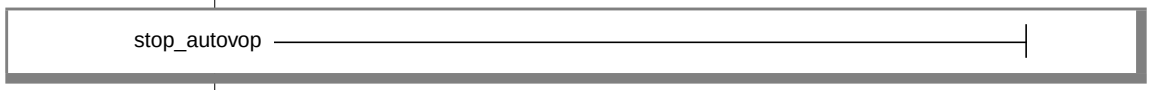
When you start **oncatlgr** with **start_oncatlgr**, **oncatlgr** directs the errors it generates to **/tmp/oncatlgr.out.*pid***, where *pid* is the process identification number of the **oncatlgr** process. The **start_oncatlgr** script prints the name of the file to where you directed output. You can stop a particular instance of **oncatlgr** using the **-f** option of the **stop_oncatlgr** command and the filename that **start_oncatlgr** reports. See "The stop_oncatlgr Script" on page 9-34 for more information on the **stop_oncatlgr** command.

Top fix most errors, simply stop and restart **oncatlgr**.

The stop_autovop Script

The **stop_autovop** script stops **onautovop** processes. Without prompting you, it stops all **onautovop** processes that you already started. When you run **stop_autovop** as **root**, it stops all **onautovop** processes for all users.

Any user can run **stop_autovop**.

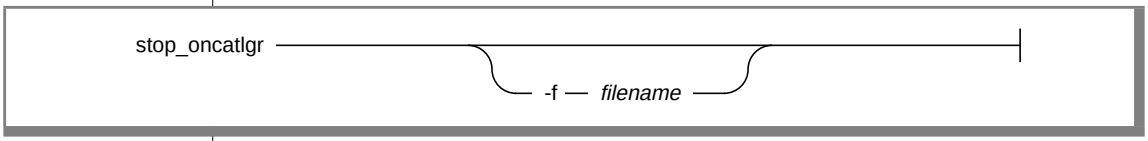


The stop_oncatlgr Script

The **stop_oncatlgr** script lets you stop **oncatlgr**. The **stop_oncatlgr** utility finds the process id for any **oncatlgr** process running and asks if you want that process stopped. When you tell it not to stop the process, **stop_oncatlgr** goes on to the next process id (when another **oncatlgr** exists). When you tell it to stop the process, **stop_oncatlgr** sends an interrupt to the process.

When you use more than one Universal Server instance on a single computer, you could encounter multiple running **oncatlgr** processes. Thus, you must take care to only stop the **oncatlgr** process associated with the correct Universal Server instance. You can use the **-f** option, or see "Which oncatlgr for Which Universal Server?" on page 4-6.

You must log in as **root** or **informix** to run **stop_oncatlgr**.



-f *filename* specifies that **stop_oncatlgr** stop only a particular **oncatlgr** process.

The *filename* must represent the output from **oncatlgr** that you wish to stop. For example, you can start **oncatlgr** with the following command:

```
% oncatlgr > dbserver7.out &
```

Then you can use the following command to stop that cataloger process:

```
% stop_oncatlgr -f dbserver7.out
```

Automating Backups Using an Event Alarm Script

When a logical log changes state, an event alarm occurs. The new event-alarm feature allows you to automatically execute backups. This feature permits you to back up logical logs without using continuous backups. In addition, you do not need to monitor the state of logical logs to know when to start an automatic backup. Informix provides a sample script to help you understand this process.

Understanding the Sample Script

The event-alarm mechanism has the facility to call only one program. That program must handle *all* event alarms that the database administrator wants to capture. To help you deal with event alarms, Informix has provided a sample script called **logevent.sh**. You can find that sample script in the **$INFORMIXDIR/etc** directory where the other ON-Archive scripts reside. You can use that script as a basis or model for automating a logical-log backup. You must edit **logevent.sh** before you can use it. The script has one limitation in that it does not know the volume sets you intend to use for log backups.

In the sample script, you can configure the following items:

- The percentage of full logs that trigger a logical-log backup
- The volume set used for the backup

When an event occurs, the database server can execute the script providing that you specify the full pathname of that script in the ONCONFIG file. When you place that script in the **$INFORMIXDIR /etc** directory, you gain an enhanced ability to administer the script.

To use an event-alarm script, perform the following steps:

1. Create a new script or configure the sample script for the correct environment.
2. Place the script in the **$INFORMIXDIR/etc** directory or any directory that you choose.
3. Use the ALARMPROGRAM configuration parameter to specify the pathname of the script in the ONCONFIG file. The pathname that you specify for the ALARMPROGRAM parameter must contain the full pathname of the sample script.

Adding an ON-Archive Activity Log to Log Archive Events

ON-Archive produces the ON-Archive activity log that contains a history of all the ON-Archive actions. You can find an example of the ON-Archive activity log in Figure 9-1 on page 9-38.

Before you start using ON-Archive, produce the ON-Archive activity log using the following steps:

1. The ON-Archive activity log can become large. Make sure the ON-Archive activity log resides in an appropriate directory for that file.
2. Make sure appropriate file permissions exist so that you can use the directory where the activity log resides.
3. Edit the **config.arc** configuration file. When you do not use the **config.arc** file, edit the file that the **ARC_CONFIG** environment variable specifies.
4. Insert an ACTIVITYLOG parameter into the specified file.
5. Make sure the ACTIVITYLOG parameter holds the full pathname of the ON-Archive activity log file. The following example shows how to set the ACTIVITYLOG parameter.

```
ACTIVITYLOG = /usr/informix/etc/onarchive.log
```

The ACTIVITYLOG configuration parameter points to the ON-Archive activity log. The ON-Archive activity log records all significant events that ON-Archive generates. The information in the ON-Archive activity log includes the beginning and end of all ARCHIVE, BACKUP, COPY, and RESTORE events.

In addition, the ON-Archive activity log contains information on dbspacesets, volume sets, volumes, and requests. The file records only commands that change the state of ON-Archive. The file does not record usage or syntax errors.

Using the ON-Archive Activity Log

Because several ON-Archive processes can run in parallel (for example, **onarchive** and **onautovop** can run simultaneously), the ON-Archive activity log conforms to a format that allows you to trace the history of the ON-Archive catalogs and follow different threads of execution.

Figure 9-1
Example of Data Within an ON-Archive Activity Log

```
Apr 19 1994 09:30 #00000000# <8734> onarchive (informix) defined dbs1and2:
define/dbspaceset=dbs1and2/dbspaces=(dbs1, dbs2)
Apr 19 1994 09:35 #00000000# <8734> onarchive (informix) defined vset1: define/vset=vset1/
access=5/device_type=tape/class=system/driver=tape
Apr 19 1994 09:37 #00000000# <9762> onarchive (informix) vset1:0001: define/volume/vset-vset1
Apr 19 1994 10:01 #00000000# <9762> onarchive (informix) created request #000000005#:
archive/dbspaceset=dbs1and2
Apr 19 1994 10:02 #00000005# <9762> onarchive (informix) begin to archive to vset1
            10:04 #00000005# <9762> processing dbs1 to vset1:0001
Apr 19 1994 10:04 #00000000# <9762> onarchive (informix) defined vset1:0002: define/volume/vset=vset1
            10:06 #00000005# <9762> processing dbs1 to vset1:000210:09 #00000005# <9762> processing
dbs2 to vset1:0002
            10:22 #00000005# <9762> failure detected: ARC-01026E Cannot close the file on tape
Apr 19 1994 10:25 #00000005# <9762> end archive: FAILED
Apr 19 1994 10:30 #00000000# <9764> onarchive (root) created request #000000006#: backup
continuous/immediate/autovop
Apr 19 1994 10:30 #00000006# <9765> onautovop (root) begin continuous backup
            10:35 #00000006# <9765> processing logfile1 to vsetlogs:0001
            10:40 #00000006# <9765> processing logfile2 to vsetlogs:0001
            10:46 #00000006# <9765> processing logfile2 to vsetlogs:0002
Apr 19 1994 11:05 #00000000# <863> ondatartr (root) begin retrieve
            11:06 #00000000# <863> processing logfile3 to ONDATARTRLOG:0001 /dev/rst0
            11:09 #00000000# <863> retrieved dbs1 #00000005# from vset1:0001,000
            11:15 #00000000# <863> retrieved dbs2 #00000005# from vset1:0002
            11:15 #00000000# <863> retrieved logfile1#00000006# from vsetlogs:000
            11:17 #00000000# <863> retrieved logfile2 #00000006# from vsetlogs:0001,0002
            11:20 #00000000# <863> retrieved logfile3 #00000863# from ONDATARTRLOG:0001
```

Figure 9-1 on page 9-38 shows several significant features:

- Date and time printed for the following events:
  - Discrete events

    ```
    Apr 19 1994 09:30 #00000000# <8734> onarchive (informix)
    defined dbs1and2: define/dbspaceset=dbs1and2/dbspaces=(dbs1, dbs2)
    ```

  - Begin marks for events that occur over a time interval

    ```
    10:30 #00000006# <9765> onautovop (root) begin continuous backup
    ```

  - End marks for events that occur over a time interval

    ```
    10:25 #00000005# <9762> end archive: FAILED
    ```

- The name of the program that generated the event
- The user ID of the person who executed the program that generated the event

  ```
  09:37 #00000000# <9762> onarchive (informix)
  vset1:0001: define/volume/vset-vset1
  ```

- Request and process IDs printed after the timestamps, that show all events appropriate to ON-Archive activity

  Hash marks (#) enclose the request ID. Greater than and less than signs (><) enclose the process ID. This information allows you to search easily for the sequence of events for a particular process, a particular request, or both.

  ```
  10:04 #00000005# <9762> processing dbs1 to vset1:0001
  ```

- The entire command issued to create, modify, or delete an ON-Archive object

  This information allows for easy re-creation of the ON-Archive catalogs and helps you understand what happens during execution events.

  ```
  10:30 #00000000# <9764> onarchive (root) created request
  #000000006#: backup continuous/immediate/autovop
  ```

- Events that take time to complete, such as ARCHIVE, BACKUP, COPY, and RETRIEVE.

  Lines that bracket those events contain the request ID and the words *begin* and *end*. The *end* events include the status of the request. In addition, error messages precede those events when failure occurs.

```
10:02 #00000005# <9762> onarchive (informix) begin to archive to vset1
10:04 #00000005# <9762> processing dbs1 to vset1:0001
Apr 19 1994 10:04 #00000000# <9762> onarchive (informix) defined
vset1:0002: define/volume/vset=vset1
10:06 #00000005# <9762> processing dbs1 to vset1:000210:09 #00000005#
<9762> processing dbs2 to vset1:0002
10:22 #00000005" <9762> failure detected: ARC-01026E Cannot close the
file on tape
Apr 19 1994 10:25 #00000005# <9762> end archive: FAILED
```

- All significant changes in state during the execution of ARCHIVE, BACKUP, COPY, and RETRIEVE events.

  These events include changing tapes or dbspaces during an archive or restore.

```
10:04 #00000005# <9762> processing dbs1 to vset1:0001
```

Chapter

10

The ON-Archive Catalog

The ON-Archive catalog tables are part of the **sysmaster** database. Unlike the system-monitoring interface tables, that are also part of the **sysmaster** database and described in the *INFORMIX-Universal Server Administrator's Guide*, the ON-Archive catalog tables are real tables.

The catalog stores all the ON-Archive requests that are not immediately executed, and information on the volume sets (vsets) and the save sets that you already defined.

Accessing the ON-Archive Catalog

You can access the ON-Archive catalog using the LIST commands in ON-Archive. However, all users can query the ON-Archive catalog tables using SQL SELECT statements. Standard users cannot execute statements other than SELECT statements on ON-Archive catalog tables.

User **informix** can execute SQL statements other than SELECT statements. However, Informix does not recommend this; updating or changing information in the catalog without using ON-Archive can adversely affect the way ON-Archive operates. When you suspect that the tables contain erroneous information, see "Repairing Catalog Inconsistencies" on page 8-64.

The ON-Archive Catalog Tables

The ON-Archive catalog consists of the following tables:

Table	Description	Page
arc_archive_event	Contains information about archive events.	page 10-5
arc_dbspace	Relates dbspaces to dbspace sets.	page 10-5
arc_dbspace_set	Describes dbspace sets.	page 10-6
arc_diskspace_mgr	Describes disk space for transit vsets.	page 10-6
arc_file	Tracks properties of archived objects.	page 10-7
arc_file_copy	Tracks location of copies of archived objects.	page 10-8
arc_pending_req	Contains pending requests.	page 10-8
arc_req_vset	Relates vsets and requests.	page 10-9
arc_request	Contains information on submitted requests.	page 10-9
arc_save_set	Contains information about save sets.	page 10-11
arc_version	Contains version information.	page 10-11
arc_vol_lock	Keeps track of locks on volumes and vsets.	page 10-12
arc_volume	Contains information about volumes.	page 10-12
arc_vset	Contains information about vsets.	page 10-14
arc_vset_user	Contains the access list for vsets.	page 10-15

In addition to these tables, ON-Archive also uses views in the **sysmaster** database to ensure data integrity when updating.

arc_archive_event

The **arc_archive_event** table contains information about archive events.

Column	Type	Description
ae_rid	INTEGER, not null	References arc_request
ae_dsid	INTEGER	References arc_dbspace_set
ae_level	SMALLINT, not null	The level of the archive
ae_timestamp	INTEGER, not null	Internal timestamp value returned by Universal Server at the time of the archive
ae_prior_rid	INTEGER	References arc_request
ae_logid	INTEGER	Oldest logical log uniqid required to restore from this archive

arc_dbspace

The **arc_dbspace** table lists each of the dbspaces and the dbspace set it resides.

Column	Type	Description
dbs_name	CHAR(18), not null	Name of dbspace
dbs_dsid	INTEGER, not null	References dbspace-set id in arc_dbspace_set

arc_dbspace_set

The **arc_dbspace_set** table describes each of the dbspace sets. The primary key is **ds_dsid**.

Column	Type	Description
ds_dsid	SERIAL	Dbspace-set id
ds_name	CHAR(18), not null	Name of the dbspace set

arc_diskspace_mgr

The **arc_diskspace_mgr** table describes the disk space allocated for transit vsets. The primary key of the table is **dsm_vid**, **dsm_vno**.

Column	Type	Description
dsm_vid	SMALLINT	Volume-set id
dsm_vno	SMALLINT	Volume number within the vset
dsm_pid	INTEGER	Process id of the user process to which the space has been allocated
dsm_spaces_alloc	INTEGER	Size of the allocated space (in 512-byte blocks)

arc_file

The **arc_file** table describes all the dbspaces and logical-log files that ON-Archive has archived. All dbspaces archived by one request form a save set. All log files archived by one automatic backup request form a save set. For a description of the relationship between logical-log files and save sets during continuous logical-log file backups, see "Performing a Continuous Backup of the Logical-Log File" on page 6-11. You can identify each dbspace or log file within the save set by a file number that ON-Archive assigns. The primary key is **file_rid**, **file_fno**.

Column	Type	Description
file_rid	INTEGER, not null	Request id
file_fno	INTEGER, not null	Unique id within the request
file_type	CHAR(1), not null	File type. 'F' is a dbspace set or log file backup.
file_name	CHAR(255), not null	File or dbspace name
file_dno	INTEGER, not null	Not used
file_nb_copies	SMALLINT, not null	Number of permanent copies created (copies on transit vsets are not counted)
file_compressed	CHAR(1), not null	'Y' when the file is compressed. The compression method and parameters are stored as part of the qualifier string in the **arc_request** table.
file_encrypted	CHAR(1), not null	'Y' when the file is encrypted. The method and parameters are stored as part of the qualifier string in the **arc_request** table.
file_creation_date	DATETIME, not null	Creation date
file_modify_date	DATETIME	Last modification date
file_expiry_date	DATETIME	Expiration date
file_comment	CHAR(80)	Optional comment about the file

arc_file_copy

The **arc_file_copy** table indicates the location of each copy of an archived dbspace set or log file. The primary key for the table is **fc_vid**, **fc_vno**, **fc_rid**, **fc_fno**.

Column	Type	Description
fc_rid	INTEGER, not null	Request id
fc_fno	INTEGER, not null	File number within the request
fc_sno	SMALLINT, not null	Sequence number, unique within each file copy
fc_vid	SMALLINT, not null	Vset id
fc_vno	SMALLINT, not null	Volume number within the vset
fc_last_sno	CHAR(1), not null	'Y' when this is the last sequence in the file copy

arc_pending_req

The **arc_pending_req** table provides a list of all unfinished requests. When a request is listed in this table, its status must be NEW, EXECUTING, or PARENT. The primary key is **pend_rid**.

Column	Type	Description
pend_rid	INTEGER, not null	The request id
pend_lock_pid,	INTEGER, not null	Process id of the process that is taking care of this request
pend_parent_rid	INTEGER	When not null, indicates the request id of the parent of this request
pend_dflt_dir_name	CHAR(255)	Pathname to the directory where the request was issued. Used as the default pathname whenever the directory is not specified in a filename.

arc_req_vset

The **arc_req_vset** table lists all the vsets required by a request. The primary key is **rv_rid**, **rv_vid**.

Column	Type	Description
rv_rid	INTEGER, not null	Request id of the request reserving a vset
rv_vid	SMALLINT, not null	Id of the vset being reserved

arc_request

The **arc_request** table describes all requests. Only requests that are not executed immediately are stored in the catalog. The primary key of the table is **req_rid**.

Column	Type	Description
req_rid	INTEGER	A unique request id, generated by ON-Archive
req_type	CHAR(2)	Request type: AR = archive request BK = backup request CO = copy request RM = remove request RT = retrieve request
req_status	CHAR(2)	Request status: NE = new request PA = parent, waiting for a child request to end CA = cancelled request FA = failed request SU = successfully completed UN = uncompleted
req_issue_date	DATETIME	Date the request was issued
req_user_node	CHAR(255)	Reserved for future use

(1 of 2)

Column	Type	Description
req_user_name	CHAR(40)	Login name of the user who issued the request
req_user_lang	CHAR(1)	Language used by the user who issued the request: E = English
req_qlf_string	CHAR(255)	Qualifier string, all of the options specified for the command
req_password	BYTE in table	Encrypted UEMK
req_former_rid	INTEGER	Imported vsets only. A new rid is assigned to avoid conflicts; this is the former rid value.
req_former_sid	SMALLINT	Former site id, used only for requests that come from imported vsets
req_expiry_date	DATETIME	Expiration date of the request
req_execution_date	DATETIME	Request execution date
req_dflt_dir_name	CHAR(255)	Default directory name of the requestor

(2 of 2)

arc_save_set

The **arc_save_set** table lists the contents of all save sets. A save set is identified by the request id of the request that created it and the vset where it resides. The primary key of the table is **svst_vid**, **svst_rid**.

Column	Type	Description
svst_rid	INTEGER, not null	Request id of the request reserving the vset
svst_vid	SMALLINT, not null	Vset id where the save set resides
svst_complete	CHAR(1), not null	'Y' when the save set is complete
svst_attached_rid	INTEGER	When not null, svst_vid must be a transit vset and svst_attached_rid indicates the request id of the request using the save set
svst_transit_date	DATETIME	When not null, indicates the expiration date of the save set
svst_nb_volumes	SMALLINT, not null	Number of volumes for this save set

arc_version

The **arc_version** table lists the version id of ON-Archive.

Column	Type	Description
av_version	CHAR(18)	Version id
av_name	BYTE in table	Name

arc_vol_lock

The **arc_vol_lock** table describes all the current locks on volumes and vsets. The primary key of the table is **vlck_vid**, **vlck_vno**, **vlck_pid**.

Column	Type	Description
vlck_vid	SMALLINT	Vset id
vlck_vno	SMALLINT	Volume number within this vset that is locked. When 0, the lock is on the vset.
vlck_rid	INTEGER	Request causing the lock; 0 when unknown
vlck_pid	INTEGER	Process id causing the lock
vlck_exclusive	CHAR(1)	'Y' when the lock is exclusive
vlck_operation	CHAR(1)	Purpose of the lock: R = read W = write D = delete

arc_volume

The **arc_volume** table describes each of the volumes known to ON-Archive. Each volume is attached to only one vset and is identified by a volume number that must be unique within the vset. The primary key of the table is **vol_vid**, **vol_vno**.

Column	Type	Description
vol_vid	SMALLINT	Vset id of the vset to which this volume belongs
vol_vno	SMALLINT	Volume number, generated by ON-Archive

(1 of 2)

Column	Type	Description
vol_max_space	INTEGER	Maximum space allowed for this volume, in 512-byte blocks. When this is 0 and the volume runs out of space, this is assigned the value of vol_used_space.
vol_used_space	INTEGER	The number of 512-byte blocks used on this volume
vol_space_exact	CHAR(1)	'Y' when it is known that the used space is an exact value
vol_full	CHAR(1)	'Y' when the volume is full
vol_nb_svst	INTEGER	Number of save-set sections in a volume
vol_nb_svst_phys	INTEGER	Number of physical save sets on the volume
vol_virtual	CHAR(255)	When the volume is a disk volume, vol_virtual specifies the pathname of the directory containing the disk volume.
vol_parameters	CHAR(20)	Volume parameters (for future use)
vol_protection	CHAR(3)	Volume protection, which can be one or more of the following: R = read access W = write access D = delete access
vol_rewind_date	DATETIME	Date of last rewind (tape volumes only)
vol_comment	CHAR(80)	Optional comment on the volume
vol_label	CHAR(6)	ANSI standard volume label
vol_remote_virtual	CHAR(255)	Remote virtual volume definition for network transit volumes

(2 of 2)

arc_vset

The **arc_vset** table describes all the vsets known to ON-Archive. The primary key of the table is **vset_vid**.

Column	Type	Description
vset_vid	SMALLINT	A unique number generated by ON-Archive
vset_name	CHAR(17)	The vset name
vset_class	CHAR(1)	The vset class: S = system U = user
vset_onsite	CHAR(1)	The location of the vset: Y = imported N = offsite U = unknown
vset_imported	CHAR(1)	'Y' when imported from another computer.
vset_foreign	CHAR(1)	'Y' when foreign (not created by ON-Archive).
vset_perm_mounted	CHAR(1)	'Y' when permanently mounted, also known as nonexpandable vset
vset_transit	CHAR(1)	'Y' when a transit vset
vset_accessibility	SMALLINT	Administrator defined accessibility level, 0 to 99
vset_owner_node	CHAR(255)	Reserved for future use
vset_dev_node	CHAR(255)	Name of the computer where the vset is or can be physically mounted
vset_dev_type	CHAR(10)	Device type on which the vset must be used
vset_dev_driver	CHAR(10)	Device driver for the physical device, DISK or TAPE

(1 of 2)

Column	Type	Description
vset_nb_volumes	SMALLINT	Number of volumes in this vset
vset_density	CHAR(4)	Density of the device (optional); valid only for tape devices
vset_location	CHAR(80)	Description of location for vset that are offsite
vset_parameters	CHAR(20)	For future use
vset_protection	CHAR(3)	Indicates vset protection and can be one or more of the following: R = read access W = write access D = delete access
vset_comment	CHAR(80)	Optional comments

(2 of 2)

arc_vset_user

The **arc_vset_user** table lists the users who can access each USER vset. The primary key of the table is **vu_vid**, **vu_user_name**.

Column	Type	Description
vu_vid	SMALLINT	Volume-set id, must be a USER vset
vu_user_node	CHAR(255)	Reserved for future use
vu_user_name	CHAR(40)	User id of the user

Chapter

11

ON-Archive Error Messages

This chapter outlines the most probable causes and solutions to errors generated during the installation and use of ON-Archive. It is intended for all users of ON-Archive. It is particularly valuable to the ON-Archive administrator.

This chapter contains the following sections:

- **oncatlgr** errors
- Operating-system errors
- Database server archive interface errors
- ON-Archive errors

"ON-Archive Errors" on page 11-13 describes the messages that ON-Archive issues when it encounters errors. ON-Archive might encounter errors in the ON-Archive catalog, in the configuration file, or as it executes commands. In addition to the error message, this section explains the likely cause of each error and suggests a course of action.

oncatlgr Errors

Almost all **oncatlgr** errors occur when **oncatlgr** receives an error back from Universal Server as it attempts to execute an SQL statement. Universal Server sends these error reports to standard output and look like the following example:

```
************************************************************
********************** ONCATLGR ERROR **********************

Error from the REQUEST SCHEDULER Module on : date-and-time

Error Description: Unable to access the database

No request assigned to that error.

LIST OF ERROR ENCOUNTERED:
    IFMX-E--25588, -25588:The apl process cannot connect to
Universal server

********************** END OF REPORT ***********************
************************************************************
```

The message number listed at the bottom of the report after the prefix IFMX-E is an INFORMIX-Universal Server message number. You can find a description of the error in the *Informix Error Messages* book, or you can use the **finderr** utility program described in that same manual.

The most common cause for an error report from **oncatlgr** occurs when Universal Server is not in on-line mode and **oncatlgr** cannot connect to the ON-Archive catalog. To solve this kind of problem, make sure Universal Server is in on-line mode and restart the cataloger.

You might also get **oncatlgr** errors when more than one **oncatlgr** process is running for a single Universal Server instance. To solve this kind of problem, stop all the **oncatlgr** processes and start only one for a single database server.

Stopping and restarting **oncatlgr** does not adversely affect the ON-Archive catalog. The catalog is logged and updates to it occur as transactions.

If you cannot correct the **oncatlgr** error with an obvious administrative action (like increasing disk space, clearing out **/tmp**, or fixing permission problems), stop **oncatlgr** and restart it. If the problem persists, contact Informix Technical Support. Support personnel need to examine the entire **oncatlgr** output file for that **oncatlgr** session.

Operating-System Errors

As **onarchive**, **onautovop**, or **ondatartr** attempt to read or write to the disk or tape volumes, they sometimes get errors back from the operating system. The operating-system error gets printed with the ON-Archive errors but retains a prefix of UNIX-E-, as shown in the following example:

```
UNIX-E-0006, No such device or address.
```

The error number that occurs after the UNIX-E- is the **errno** value that ON-Archive detected. The message that ON-Archive prints is a short interpretation of the UNIX **errno** value. To determine the cause of the error, look in your operating-system documentation about the type of device that was being used at the time of the error. Common errors include trying to use pathnames for which user operating-system permission does not exist, forgetting to mount a tape, or trying to use a tape drive already in use.

If you cannot operationally correct the error, contact Informix Technical Support. Support personnel need to see the exact text of the error messages that were produced. They also need to know the series of events that led up to the errors.

Database Server Archive-Interface Errors

During the processing of an archive, a logical-log file backup, or a restore, **onarchive**, **onautovop**, and **ondatartr** communicate with Universal Server through a special interface known as the archive API. You can get errors back from Universal Server through this interface. ON-Archive distinguishes such errors with the prefix IFMX-E-API, as in the following example:

```
IFMX-E-API, There are no DBSpaces or BLOBSpaces to recover.
Check the status by using onstat -d.
```

These self-explanatory errors do not have Informix error numbers associated with them. You cause the most common errors you attempt to perform archives, backups, or restores when Universal Server is not in the correct mode.

If you cannot operationally correct the error, contact Informix Technical Support. Support personnel need to see the exact text of the error messages that were produced. They also need to know the series of events that led up to the errors.

ON-Archive Errors

This section describes the ON-Archive errors that **onarchive**, **ondatartr**, or **onautovop** generate. This section is divided into the following parts:

- About ON-Archive errors
- ON-Archive configuration errors
- ON-Archive errors
- An alphabetical list of ON-Archive errors

About ON-Archive Errors

This section contains useful information for reading and interpreting ON-Archive error messages.

Variables in Messages

Some messages contain a variable that a value replaces when ON-Archive displays the message. The % character indicates the variable in the message, followed with a letter that specifies the variable type. For example, in the following error message,

```
ARC-01011E      Ambiguous command abbreviation %s.
```

When the error message displays, a string of text replaces the variable %s. The actual error message looks something like the following example:

```
ARC-01011E      Ambiguous command abbreviation EX.
```

Multiple Error Messages

Occasionally, you might receive multiple error messages; a series of related error messages that result from the execution of your command.

Internal Errors and Signals

ON-Archive performs extensive self-checking to determine internal consistency errors. When ON-Archive finds an internal error, ON-Archive displays the following messages:

```
ARC-E-1576  Unexpected internal fatal error has occurred.
ARC-I-1587  Internal check failure code: %s
ARC-I-1578  Consult your /tmp/sysfail.%s file.
```

A system failure occurs when a UNIX signal occurs. In this event, ON-Archive displays the following messages:

```
ARC-E-1116  System failure.
ARC-I-1577  Program will be aborted to produce core dump.
ARC-I-1578  Consult your /tmp/sysfail.%s file.
ARC-E-1307  Unix signal received. Signal number = %d.
```

Also, that failure produces a core dump of ON-Archive. This core dump might help the support person identify the cause of the system failure.

When you experience either of these problems, report it as a software problem to Informix Technical Support. Please prepare to describe exactly what operation you were executing and when the messages appeared. As well, prepare to describe the contents of the file **/tmp/sysfail.***pid*, where *pid* is the process id of the ON-Archive process involved in the error. This file contains details about exactly where the internal problem was detected and is required to locate the source of the problem.

Calling for Technical Support

Some error messages recommend that you call your Informix Technical Support representative and report the error. When you call, please have the following information available:

- The hardware, operating system, and release number of the operating system on which ON-Archive is running
- The version number of Universal Server

- The exact error code or messages that appeared
- A description of the problem, including the operation that was being executed when the error occurred

ON-Archive Configuration Errors

Configuration error messages occur when the cataloger reads the ON-Archive configuration file, **config.arc**. The cataloger reads the file each time you start it with the **start_oncatlgr** command. When you start an ON-Archive session you enter the command **onarchive** from the operating system, and you also read the configuration file.

When you receive a configuration error message, stop the cataloger with the **stop_oncatlgr** command and then restart it after you make the appropriate changes to the configuration file.

Where Are ON-Archive Configuration Errors Displayed?

ON-Archive displays configuration error messages on your terminal when you enter either of the ON-Archive interfaces.

Parts of a Message

A typical configuration error message looks like this:

```
ARC-CONFIG,Language must be on the first line.
```

Each configuration error message is divided into two parts as shown in the following list:

- The prefix (`ARC-CONFIG`)

  The prefix indicates the source of the message.
- The message text (`Language must reside on the first line.`)

  The message text describes the error.

For each error, the error message list gives the probable cause and most-likely solution.

List of Configuration Errors

ARC-CONFIG, Configuration file not found in **$INFORMIXDIR/etc**.

Cause: A file named **config.arc** could not be accessed or found in the **$INFORMIXDIR/etc** directory.

Action: Place a file with configuration parameters described in Chapter 3, "Configuring ON-Archive," in **$INFORMIXDIR/etc/config.arc**.

ARC-CONFIG, Database name too long, - Max is 30 characters.

Cause: The database name specified in the configuration file is longer then 30 characters.

Action: Shorten the database name.

ARC-CONFIG, Error and message file undefined for all language.

Cause: Both the error file and the message file are not defined for any languages.

Action: Supply the definition of both the error file and the message file in the configuration file.

ARC-CONFIG, File %s not found.

Cause: The specified file displayed in the error message could not be found in the **$INFORMIXDIR/msg** directory.

Action: Modify the file specification as found in the ON-Archive configuration file, **config.arc**, so that it points to the valid file location.

ARC-CONFIG, File(s) corresponding to **config.language** not found.

Cause: One or more of the required files for the language specified in the configuration file have not been found.

Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-CONFIG, Language must be on the first line.

Cause: The first noncommented line of the configuration file does not specify a supported language.

Action: Modify the first noncommented line of the configuration file so that it specifies a supported language.

ARC-CONFIG, List of value not permitted.

Cause: A list of values was specified in the configuration file for a parameter that does not support lists.

Action: Modify the configuration file so that the parameter that contained a list of values only has one value.

ARC-CONFIG, Missing catalog message file name.

Cause: The configuration file does not include the specification of a catalog message filename.

Action: Modify the configuration file so that a catalog message filename is specified.

ARC-CONFIG, Missing device name.

Cause: The configuration file does not include a specification of a device name. At least one device must be specified in the configuration file.

Action: Modify the configuration file so that a device name is specified.

ARC-CONFIG, Missing equal sign '='.

Cause: The configuration file is missing an equal sign (=) on one of its specification lines.

Action: Supply the missing equal sign in the configuration file.

ARC-CONFIG, Missing file spec. after equal sign.

Cause: The configuration file does not include a file specification after the equal sign.

Action: Modify the configuration file so that a file specification follows an equal sign.

ARC-CONFIG, Missing file language.

Cause: The configuration file does not include the specification of a language.

Action: Modify the configuration file so that it includes the specification of a language.

ARC-CONFIG, Missing language, (English or Francais).

Cause: The configuration file does not include the specification of a language.

Action: Modify the configuration file so that it includes the specification of a language.

ARC-CONFIG, Missing parameter.

Cause: The configuration file is missing a parameter.

Action: Modify the configuration file so that the missing parameter is added.

ARC-CONFIG, Missing qualifier.

Cause: The configuration file is missing a qualifier.

Action: Modify the configuration file so that the missing qualifier is added.

ARC-CONFIG, Unknown parameter.

Cause: The configuration file includes an unknown parameter.

Action: Modify the configuration file so that the unknown parameter is removed.

ARC-CONFIG, Unknown privilege type.

Cause: The configuration file includes a privilege type that is unknown.

Action: Modify the ON-Archive configuration file, **config.arc**, so that the privilege type is valid. Refer to Chapter 3, "Configuring ON-Archive."

ARC-CONFIG, 4th character of ACS device type must be '1', '2', '3' or '4'.

Cause: The fourth character of the specification for an ACS device is not valid.

Action: Change the specification of the ACS device so that the fourth character of the device type is either a '1', '2', '3', or '4'.

ARC-CONFIG, The pool id must be in the range 1 to 255.

Cause: The pool id specified is not valid.

Action: Change the pool id specified so that it is in the range 1 to 255.

ARC-CONFIG, Unknown protocol.

Cause: The protocol specified is not recognized by ON-Archive.

Action: Specify a protocol that is recognized by ON-Archive. For a complete list of the protocols supported, contact Informix Technical Support.

ARC-CONFIG, Unknown security mode.

Cause: The security mode specified is not recognized by ON-Archive.

Action: Change the security mode to one that is supported.

ARC-CONFIG, The nb_disk_space_extent must be in the range 1 to 100.

Cause: The value for NB_DISK_SPACE_EXTENT is not in the supported range of values.

Action: Modify the value specified for NB_DISK_SPACE_EXTENT to be within the range 1 to 100.

ARC-CONFIG, The option %s is ignored for Informix users.

Cause: This is informational only. The specified parameter is valid for the Archive*SQL product but is ignored by ON-Archive. This occurs when Archive*SQL and ON-Archive share the same **config.arc** file.

Action: None.

ON-Archive Errors

This section lists error messages that **onarchive**, **onautovop**, and **ondatartr** generate. Some of these messages are informational and are perfectly normal. For example, when you try to query the catalog to locate a file you never archived, you receive an informational message indicating that the file does not reside in the catalog. This message does not indicate that an error occurred, but rather that no data stored in the catalog could satisfy your query.

Where Are ON-Archive Error Messages Displayed?

ON-Archive error messages are displayed differently depending on the context. When you interactively execute an ON-Archive command (with **onarchive**), the error message is displayed on your terminal. All generated error messages are also written to a log file (When you specify LOG for the command) and get sent to the owner of the request using electronic mail (When you specify NOTIFY for the command).

When you use the Menu interface, press CTRL-E to recall the last set of error messages.

When **onautovop** executes requests, errors are mailed to the owner of the request.

Parts of a Message

A typical ON-Archive error message looks like this:

```
ARC-E-01040     File not found.
```

Each error message and code is divided into four parts.

- The prefix (`ARC`)

  The prefix indicates the source of the message.
- The type of message (`E`)

  Four types are currently supported:

E	Error message
F	Fatal error message
I	Informational message
W	Warning message

- The message number (`01040`)

  Each message is assigned a sequential number. The specific error number associated with a message has no special meaning.
- The message text (`File not found`)

  The message text describes the problem.

For each error, the probable cause and most likely solution are given.

The errors listed here are in a different format than when ON-Archive returns them. This chapter places the type of the message after the message number, without a separating hyphen.

List of ON-Archive Errors

ARC-01000F Undefined format '%s'.

Cause: An attempt to display an unknown message was discovered.

Action: Check that the appropriate version of the format file (**fmt_E.dat** or **fmt_F.dat**) is in the **$INFORMIXDIR/msg** directory (or filename specified in **config.arc**).

ARC-01001F Cannot close format file.

Cause: The operating system returned an error during the close of the format file.

Action: Make sure the format file exists and can be read by ON-Archive users.

ARC-01002F System routine failure.

Cause: ON-Archive makes calls to the operating system to get the current user id and group information. The operating system returned an error on a system call.

Action: Verify that the user id being used to execute **onarchive** is properly defined.

ARC-01004F Format file not found.

Cause: ON-Archive cannot find the format file that contains informational messages.

Action: Check that the **fmt_E.dat** (or the MESSAGE file specified in **config.arc**) is accessible from **$INFORMIXDIR/msg**.

ARC-01005F Cannot read format file.

Cause: ON-Archive received an error while trying to read the format file.

Action: Check that the **fmt_E.dat** (or the MESSAGE file specified in **config.arc**) is accessible from **$INFORMIXDIR/msg**. Check that the file has the expected ASCII text in it.

ARC-01009E Illegal default setting.

Cause: One of the default qualifiers in the **oper_deflt.arc** file is unknown to ON-Archive.

Action: Check that **oper_deflt.arc** (or the DEFAULT file specified in **config.arc**) is accessible from **$INFORMIXDIR/msg**. Check that all the words used in **oper_deflt.arc** are known ON-Archive qualifiers.

ARC-01010E Allocation failure.

Cause: An error occurred while trying to allocate additional memory from your operating system (**onarchive** uses the UNIX **malloc** command).

Action: Inform your system administrator so he or she can determine the nature of the problem.

ARC-01011E Ambiguous command abbreviation %s.

Cause: The command abbreviation specified was not unique; therefore, ON-Archive could not determine what the full command name was.

Action: Lengthen the command abbreviation so that it is no longer ambiguous.

Example: EX can be the abbreviation for EXit or EXecute.

ARC-01012E Ambiguous qualifier abbreviation %s.

Cause: The qualifier abbreviation specified was not unique; therefore, ON-Archive could not determine what the full qualifier name was.

Action: Lengthen the qualifier abbreviation so that it is no longer ambiguous.

Example: COM can be the abbreviation for COMment or COMmand.

ARC-01013E Cannot archive on a transit vset.

Cause: A vset defined as transit was specified as the final destination of an archive command. Transit vsets can only be specified as intermediary destinations for unattended operations (using the TRANSIT qualifier).

Action: Specify another vset as the final destination (using the VSET qualifier) of the archive command.

ARC-01016E Cannot set protection to Write access when imported.

Cause: Write protection was specified while defining a vset as imported.

Action: Remove the write access (W) from the PROTECTION qualifier and resubmit the request.

ARC-01017E A transit vset cannot be copied.

Cause: A COPY command specified a vset that was defined as transit as its source. Transit vsets cannot be copied because they are only used for temporary storage.

Action: Either select a vset that is not defined as transit as the source of the copy command, or move the data from the transit vset to its destination vset by executing the appropriate child-copy request.

ARC-01018E Request already completed or cancelled.

Cause: An EXECUTE operation has been specified against a request id that has either completed or has been cancelled.

Action: Choose a request id that has neither been completed nor cancelled. Or, modify the request so that its status is NEW, using the MODIFY / COMMAND command.

ARC-01019E Cannot open user notice file.

Cause: The notice file is placed in the **$HOME** directory of the user id executing **onarchive**. If **onarchive** does not have access to write into that directory, it attempts to write the notice file into **/tmp**. If that fails, **onarchive** displays this error.

Action: Check the setting of **$HOME** and also check the home-directory permissions.

ARC-01020E Input vset read protected.

Cause: The requested operation includes a read against a vset that is read protected.

Action: Modify the protection of the vset using the command MODIFY / VSET or choose a different vset.

ARC-01022E Output vset write protected.

Cause: The operation specified includes a write operation against a vset that is write protected.

Action: Modify the protection of the vset using the command MODIFY / VSET or choose a different vset.

ARC-01023E Catalog is corrupted.

Cause: The catalog has been corrupted.

Action: First, run the **arc_purge.sql** script using DB-Access, but only if no archive commands are executing. Then try again to do whatever caused the error. If a problem still occurs, find out what the catalog is missing and recatalog that information using the CATALOG command.

ARC-01024E Cannot close the file on disk.

Cause: While attempting to close a disk archive or backup file, the operating system returned an error.

Action: Check the error number returned with the operating-system error message to determine the nature of the problem.

ARC-01026E Cannot close the file on tape.

Cause: While attempting to close the tape device, **onarchive** received an operating-system error.

Action: Check the error number with the operating-system error message to determine the nature of the problem.

ARC-01027E Data is corrupted (modified).

Cause: During the restoration of data, a corrupt data file was discovered.

Action: Reexecute the retrieve request to confirm that the data file is corrupt. If it is corrupt, try to retrieve it using a copy of the file or a previous version. If that too fails, contact your Informix Technical Support representative.

ARC-01032E Device too long in file specification.

Cause: The device portion of the file specification is invalid under your operating system.

Action: Modify the device portion of the file specification such that it is compatible with your operating system.

ARC-01035E Duplicate records.

Cause: A DEFINE command has been specified using a name that already exists in the catalog.

Action: Modify the name specified so that it is unique within the catalog, or delete the existing object that uses the same name.

ARC-01038E File extension too long in file specification.

Cause: The file-extension portion of the file specification is not compatible with your operating system.

Action: Modify the file extension portion of the file specification so that it is compatible with your operating system.

ARC-01039E Fgets function failure.

Cause: An attempt to read from a disk volume failed after the volume was successfully opened.

Action: Check the disk volume for corruption.

ARC-01040E File not found.

Cause: The command that was entered attempted to locate files for a request. No files were located in the archive catalog.

Action: Verify that an operating-system file exists where you specified. Recatalog the volumes if necessary.

ARC-01041E File not in save-set.

Cause: A save set is being read, but the expected file was not located in the save set. The ON-Archive catalog might be out of synch with what is actually on tape or disk.

Action: Check that the save sets and files being retrieved actually exist on the media. If not, update the ON-Archive catalog either by removing any defunct requests or by recataloging the volume.

ARC-01042E Stat can't return the file attributes.

Cause: ON-Archive does an operating-system stat call to get information about disk files. The stat call returned an error.

Action: Check for disk corruption.

ARC-01043E File name too long in file specification.

Cause: The filename portion of the file specification is not compatible with your operating system.

Action: Modify the filename portion of the file specification so that it is compatible with your operating system.

ARC-01044E Fseek function failure.

Cause: While attempting an fseek operation on a disk file, ON-Archive received an error result.

Action: Check for disk corruption.

ARC-01045E Cannot open disk file %s.

Cause: An attempt to open the specified disk file failed.

Action: Verify that the specified file is accessible to the user id that was executing **onarchive**.

ARC-01047E Insufficient privilege.

Cause: You are trying to execute a command for which you do not have sufficient privileges; for example, trying to execute DEFINE / VSET if you are not an operator and ON-Archive is running in OPERATOR privilege mode.

Action: Change the privilege mode that ON-Archive is running in, or have a user with sufficient privilege execute the command.

ARC-01048E Invalid blocksize.

Cause: The block size specified is not supported by the operating system.

Action: Reduce the block size specified to one supported by ON-Archive under your operating system.

ARC-01049E Invalid close bracket in file specification.

Cause: A closing bracket (]) was found in an invalid position in the file specification.

Action: Omit the invalid closing bracket.

ARC-01050E Invalid command.

Cause: The command entered is not supported.

Action: Enter another command.

ARC-01051E Invalid colon in file specification.

Cause: A colon (:) or semicolon (;) was found in an invalid position in the file specification.

Action: Omit the invalid colon or semicolon.

ARC-01052E Invalid compression method or parameter syntax error.

Cause: The compression method specified is unknown, or there is an error in the syntax of the parameters supplied to the compression algorithm.

Action: Specify a valid compression method and use the proper syntax to specify any parameters. Use the LIST/METHOD command to display all supported compression methods.

ARC-01054E Invalid device in file specification.

Cause: The device portion of the file specification does not exist or is not compatible with your operating system.

Action: Modify the device portion of the file specification to a valid device.

ARC-01055E Invalid directory in file specification.

Cause: A VIRTUAL pathname for a disk volume does not exist.

Action: Create the directory pathname for the disk volume and try the request again.

ARC-01056E Invalid encryption method or parameter syntax error.

Cause: The encryption method specified is unknown or there is an error in the syntax of the parameters supplied to the encryption algorithm.

Action: Specify a valid encryption method and use the proper syntax to specify any parameters. Use the LIST/METHOD command to display all supported encryption methods.

ARC-01057E Invalid file extension in file specification.

Cause: The file- extension portion of the file specification is invalid under your operating system.

Action: Modify the file-extension portion of the file specification so that it is compatible with your operating system.

ARC-01058E Invalid file name in file specification: %s

Cause: The filename specified does not exist.

Action: Change the filename to one that exists or use a wildcard character so that the file specification matches at least one file.

ARC-01059E Invalid file version in file specification.

Cause: The file version number portion of the file specification is invalid under your operating system.

Action: Modify the file version number portion of the file specification such that it is compatible with your operating system.

ARC-01061E Invalid language specified.

Cause: The language specified has not been defined for your installation of ON-Archive.

Action: Verify that the specified language has not been defined for your installation of ON-Archive.

Verify that the language you desire is available. Contact your local sales representative for more information about support for the language of your choice.

ARC-01062E Invalid node name in file specification.

Cause: The node-name portion of the file specification does not exist or is not compatible with your operating system.

Action: Modify the node name to one that exists and is compatible with your operating system.

ARC-01063E Invalid open bracket in file specification.

Cause: An opening bracket ([) was found in an invalid position in the file specification.

Action: Omit the invalid opening bracket.

ARC-01064E Invalid parameter.

Cause: The value of a parameter specified for one of the qualifiers in the command does not have an appropriate value.

Action: Check the documentation specified for the correct parameter value ranges for each qualifier and enter the command again with proper values.

ARC-01065E Invalid parameter type “%s”.

Cause: The value for the specified parameter is not of the correct type. For example, BLOCKSIZE = “abc” would be an invalid parameter type because BLOCKSIZE is expecting a numeric value.

Action: Check the documentation for the correct type of value for the parameter and enter the command again with proper values.

ARC-01066E Invalid period position in file specification.

Cause: The position of one or more of the periods in your file specification is not compatible with your operating system.

Action: Modify the file specification so that it is compatible with your operating system.

ARC-01067E Invalid semi-colon in file specification.

Cause: A semicolon (;) was found in an invalid position in the file specification.

Action: Omit the invalid semi-colon.

ARC-01072E Parameter list not permitted - Single parameter only.

Cause: One of the specified qualifiers does not accept a list of parameters. Only one parameter is allowed.

Action: Remove the multiple parameters from the parameter list.

ARC-01073E Missing /SINCE or /BEFORE.

Cause: The specified command requires either the /SINCE or /BEFORE qualifier. This commonly occurs when using the ARCHIVE/FILE command where the qualifier /MODIFIED, /CREATED, or /EXPIRED is used.

Action: Either remove the /MODIFIED, /CREATED, or /EXPIRED qualifier from the command, or add the missing /SINCE or /BEFORE qualifier.

Example: ARCHIVE/DBSPACESET=(*)/CREATED

ARC-01074E Missing bracket.

Cause: A bracket was missing in the specification of a file or directory.

Action: Supply the missing bracket.

ARC-01075E Missing close bracket in file specification.

Cause: A closing bracket (]) was missing in the specification of a file or directory.

Action: Supply the missing bracket.

ARC-01076E Missing command.

Cause: The command line entered does not include an ON-Archive command.

Action: Supply the missing command at the beginning of the command line.

ARC-01077E Missing comma (,) in parameter list.

Cause: One or more of the parameters in the parameter list are not separated by commas (,).

Action: Insert the missing commas (,) in the parameter list.

ARC-01078E Missing device name in file specification.

Cause: The device-name portion of the file specification is missing.

Action: Supply the missing device name in the file specification.

ARC-01079E Double quote (“) not properly ended.

Cause: The command included an opening double quote (“) but did not include a closing double quote (”).

Action: Insert the missing double quote (“).

ARC-01081E Missing node name in file specification.

Cause: The node-name portion of the file specification is missing.

Action: Supply the missing node name in the file specification.

ARC-01082E Missing open bracket for parameter list.

Cause: The specified parameter list does not begin with an open bracket [(].

Action: Insert the missing open bracket [(] at the beginning of the parameter list.

ARC-01083E Missing parameter for qualifier %s.

Cause: The command specified is missing one or more parameters. This is typically caused by placing two or more commas (parameter separators) consecutively in the command or forgetting to supply a parameter value for a qualifier, which requires one.

Action: Remove the extra parameter separators or add the missing parameters.

Example: ARCHIVE / DBSPACESET=(X,Y,,,,Z)

ARC-01084E Missing parenthesis.

Cause: A parenthesis was missing in the specification of a file or directory.

Action: Supply the missing parenthesis.

ARC-01085E Slash (/) not followed by a qualifier name.

Cause: The command included a slash (/), but it was not followed by a qualifier name.

Action: Remove the extra slash (/) from the command or supply the missing qualifier name.

ARC-01086E Missing qualifier %s.

Cause: One of the mandatory qualifiers of a command was not specified.

Action: Supply the missing qualifier. If you are not sure which qualifier is missing, refer to the reference material for the command or on-line Help facility.

ARC-01087E Missing slash (/) before qualifier name.

Cause: A qualifier name specified in the command was not preceded by a slash (/).

Action: Supply the missing slash (/) before the qualifier name or remove the qualifier name that does not begin with a slash.

ARC-01088E Save-set header not found in the save-set.

Cause: ON-Archive could read the volume header but could not read the save-set header.

Action: Verify that the right tape or disk volume is being used. The tape or disk might have become corrupted. Use a copy.

ARC-01089E Mount failure.

Cause: An error occurred while mounting the specified device.

Action: Verify that the specified device is on-line.

ARC-01096E Node name too long.

Cause: The node-name portion of the file specification is too long for your operating system.

Action: Modify the node-name portion of the file specification so that it is compatible with your operating system.

ARC-01098I Not an onarchive volume.

Cause: The attempted operation against the specified volume could not be completed because the volume was not created by ON-Archive.

Action: Select another volume for the operation or initialize the volume using ON-Archive.

ARC-01099E Item not found in catalog.

Cause: The requested information is not in the archive catalog.

Action: Do LIST commands to see what the archive catalog lists in its tables.

ARC-01101E User is not the owner of the specified request.

Cause: The command entered could not be completed because the user executing the command was not suitably privileged. That is, the user was not the owner of the request, nor did the user have operator privilege.

Action: Have either the owner of the request or a user with operator privilege execute the command.

ARC-01103E	Missing qualifier in /VOLUME.
Cause:	The VOLUME qualifier is missing a parameter that is required within the context it is used.
Action:	Supply the missing parameter. Refer to the reference material for the command or on-line Help facility for the exact syntax of the specified command.
ARC-01104E	Save-set open failure.
Cause:	An error occurred trying to open a save set.
Action:	Verify that the user who entered the request has read access (for a retrieve command) or read and write access (for an archive command) to the virtual directory specified by the DEFINE/VOLUME command.
ARC-01105E	Qualifier list too long (?).
Cause:	The command is too long to store in the archive catalog.
Action:	Make sure that this is a valid command. Remove duplicate qualifiers.
ARC-01108E	Default file read failure.
Cause:	ON-Archive detected an error during a read of the **oper_deflt.arc** file.
Action:	Verify that the **oper_deflt.arc** file is in **$INFORMIXDIR/etc** and is accessible to the user id executing **onarchive**.
ARC-01110E	Cannot rewind tape volume.
Cause:	ON-Archive attempted to rewind the tape volume, using **ioctl** commands, but detected an error result.
Action:	Use the operating-system error number to determine why the rewind does not occur on the tape device.

ARC-01112E Cannot skip to the next save-set or no more save-sets.

Cause: ON-Archive attempted to skip past one save set to the next, using **ioctl** commands to skip file marks, but detected an error.

Action: Use the operating-system error number to determine why the directives to skip file marks failed.

ARC-01113E Sub-directory too long in file specification.

Cause: The subdirectory portion of the file specification is too long for your operating system.

Action: Modify the subdirectory portion of the file specification so that it is compatible with your operating system.

ARC-01114E Save-set buffer (read or write) not allocated.

Cause: A memory allocation failed, preventing **onarchive** from allocating its read or write buffers.

Action: Have your system administrator determine why the system does not allow a UNIX **malloc** to allocate memory to this process.

ARC-01116E System failure.

Cause: A fatal error occurred that caused the operating system to issue a signal.

Action: Examine the **/tmp/sysfail.*pid*** file to see the details of the error, where pid is the process id of the ON-Archive process involved in the error. Contact your Informix Technical Support representative and report the problem.

ARC-01118E Number of vsets not equal to the number of copies.

Cause: The number of copies specified does not match the number of vsets specified in the ARCHIVE command.

Action: Reduce the number of copies specified or increase the number of vsets specified for the ARCHIVE command, until both values are identical.

Example: ARCHIVE / DBSPACESET=rootset / COPIES=3- / VSET=(FIRST,SECOND)

ARC-01119E Too many parameters.

Cause: A qualifier has more parameter values than it is allowed.

Action: Check the syntax of the command qualifiers and only enter as many parameter values as are allowed for each qualifier.

ARC-01120E Specified transit vset is not defined as transit.

Cause: The vset specified as transit (using the TRANSIT qualifier) has not been defined as type transit.

Action: Change the vset specified to one of type transit.

ARC-01121E Undefined device type (see CONFIGURATION file).

Cause: The specified device does not match the list of available devices in the ON-Archive configuration file, **config.arc**.

Action: Verify that the specified device exists in the configuration file. If the error message occurred during a DEFINE / VOLUME command, also verify that the device is valid for the DEVICE_TYPE specified during the definition of the vset. This can be verified by using the LIST / VSET command.

ARC-01122E Undefined volume %s.

Cause: A command included a reference to a volume of a specified vset that is not defined in the catalog.

Action: Modify either the volume or vset specified so that the reference is to a known volume, or define the appropriate volume of the specified vset.

ARC-01123E Undefined vset.

Cause: A command included a reference to a vset that is not defined in the catalog.

Action: Modify the command so that it refers to a known vset or define the appropriate vset.

ARC-01124E Unexpected closing bracket ()).

Cause: A closing bracket [)] was found that does not have a corresponding opening bracket [(].

Action: Remove the extra closing bracket [)] or supply the missing opening bracket [(].

ARC-01125E Unexpected comma (,).

Cause: One of the commas entered in the specified command is in an unexpected place.

Action: Verify the positioning of all the commas in your command.

Example: ARCHIVE / DBSPACE=(*,)

ARC-01126E Unexpected parameter with qualifier %s.

Cause: An unexpected parameter was included with the qualifiers.

Action: Remove the unexpected parameter. If you are not sure which parameter was not supported, refer to the documentation for the command, or to the on-line Help facility.

ARC-01128E Qualifier %s not allowed with this command.

Cause: An unexpected qualifier was included in a command.

Action: Remove the unexpected qualifier. If you are not sure which qualifier was not supported, refer to the documentation for the command, or to the on-line Help facility.

ARC-01129E Unknown command %s.

Cause: The command entered was not recognized.

Action: Enter a command that is recognized. If you are not sure what the correct command is, refer the documentation for the command, or to the on-line Help facility.

ARC-01132E Unknown qualifier %s.

Cause: The specified qualifier was not recognized.

Action: Check the spelling of the qualifier. If you are not sure of the correct usage for the qualifier, refer to the documentation for the command, or to the on-line Help facility.

ARC-01136E Wrong volume mounted.

Cause: The volume mounted on the specified device belongs to the correct vset but is not the correct volume.

Action: Mount the correct volume of the required vset.

ARC-01137E Wrong vset name.

Cause: The volume mounted on the specified device does not belong to the correct vset.

Action: Mount the correct volume of the required vset.

ARC-01139E User name too long.

Cause: The username specified is longer than the maximum length supported.

Action: Shorten the length of the user name.

ARC-01140E No authorization to use the specified vset.

Cause: The user is trying to access a vset for which he or she does not have access permission.

Action: Use a vset for which you have access.

ARC-01142E Volume(s) write protected.

Cause: An attempt was made to write to a volume that has been write protected.

Action: Modify the protection on the volume, using the MODIFY / VOLUME command.

ARC-01143E Volume-set protected against deletion.

Cause: An attempt was made to delete a vset that is delete protected.

Action: Modify the protection of the vset using the MODIFY / VSET command.

ARC-01145E Virtual volume must be on a disk vset.

Cause: The VIRTUAL qualifier of the DEFINE / VOLUME command only applies to vsets that are defined using the DRIVER=DISK qualifier.

Action: Remove the VIRTUAL qualifier or change the vset name to a vset defined using the DRIVER=TAPE qualifier.

ARC-01147E Vset write protected.

Cause: Cannot write to the specified vset because the write protection has been enabled.

Action: Either use a different vset, or modify the protection of this vset.

ARC-01148E Tape corrupted, cannot read or write on it.

Cause: After opening and verifying the tape, **onarchive** attempted to read the tape and got an error.

Action: The tape is probably corrupt. Use a different copy of the save set.

ARC-01150E Write tape mark failed.

Cause: ON-Archive attempted to write a tape mark to the tape and detected an error.

Action: Use the operating-system error number to determine why the tape mark could not be written.

ARC-01154E Tape vset cannot be transit.

Cause: A TAPE-based vset was attempted to be defined as type transit.

Action: Remove the TRANSIT qualifier from the DEFINE/VSET command or specify a vset that was defined using the DRIVER=DISK qualifier.

ARC-01155E A transit vset must be permanently mounted.

Cause: A transit vset was attempted to be defined as] NOPERMANENTLY_MOUNTED.

Action: Remove the NOPERMANENTLY_MOUNTED qualifier or remove the TRANSIT qualifier.

ARC-01156E Fwrite in the save-set file failed.

Cause: One of the reasons for receiving this message is if during an archive operation on a disk vset, the user who entered the archival request did not have enough disk quota to complete the operation.

Action: Increase the users disk quota on the appropriate device.

ARC-01157E Fputs function failure.

Cause: An attempt to write to a disk file resulted in an error.

Action: Use the other error messages to determine why the error occurred.

ARC-01159E /TRANSIT must be used as toggle.

Cause: In this context, TRANSIT must not have a parameter value.

Action: Resubmit the command with the correct syntax.

ARC-01160E Missing username in /USER qualifier.

Cause: The USER qualifier requires a username when used in the current context.

Action: Supply the missing username.

ARC-01162E Node not permitted in /VIRTUAL qualifier.

Cause: The VIRTUAL qualifier does not permit a node name as part of the directory specification.

Action: Remove the node name from the directory specification.

Example: DEFINE/VSET=TEST/VOL/MAX_SPACE=10000/VIRTUAL=(node@/filesystem/directory)

ARC-01163E Missing directory in /VIRTUAL qualifier.

Cause: The VIRTUAL qualifier is missing the directory name from its directory specification.

Action: Supply the missing directory specification.

Example: DEFINE/VSET=TEST/VOL/MAX_SPACE=10000/VIRTUAL=(/filesystem)

ARC-01164E Bad accessibility range, must be between 0-99.

Cause: The specified accessibility parameter was not between the values 0 and 99.

Action: Modify the accessibility so that it is between 0 and 99.

Example: MODIFY/VSET=TEST/ACCESSIBILITY=200

ARC-01165E Undefined device (see CONFIGURATION file).

Cause: The physical device specified was not found in the ON-Archive configuration file, **config.arc**.

Action: Either modify the device to one that is specified in the configuration file, or add the desired device to the configuration file.

ARC-01166E The qualifier /VSET (volume set name) is missing.

Cause: The qualifier VSET is mandatory for the command entered.

Action: Supply the missing VSET qualifier.

ARC-01168E No user list permitted.

Cause: A list of user names is not permitted in the current context.

Action: Specify only one username.

ARC-01169E No volume number list permitted.

Cause: The command entered does not support a list of volumes nor the (*) wildcard. Only one volume is permitted.

Action: Remove the list of volume numbers and specify only one volume.

Example: CATALOG / VSET=TEST / VOL=(1,2)

ARC-01170E Invalid date format %s.

Cause: The date format specified is not valid.

Action: Modify the date format so that it is valid. Refer to the documentation for the command, or to the on-line Help facility for more information.

Example: ARCHIVE / DBSPACESET=rootset / WAIT=T:1234567890

ARC-01171E Date parameter list not permitted.

Cause: A list of dates is not permitted in the current context.

Action: Specify only one date parameter.

ARC-01172E / VIRTUAL cannot be changed when volume is not empty.

Cause: The VIRTUAL qualifier of a volume cannot be changed unless the entire volume is empty.

Action: Empty the volume by using the DELETE command to remove the save sets that reside in the volume.

Warning: *This action will remove data from the catalog.*

ARC-01173E Request %s is already terminated or PARENT.

Cause: The attempted operation has already completed.

Action: No action.

ARC-01174E Undefined request %s.

Cause: A reference was made to an undefined request id.

Action: Modify the request id to one that exists in the catalog.

ARC-01175E fread function failed.

Cause: A read on a disk file failed after the file had been opened and positively identified.

Action: Check for disk corruption.

ARC-01177E Not enough VSETS for the requested number of copies.

Cause: An ARCHIVE request specified more copies than vsets defined in the catalog as accessible to the current user.

Action: Specify fewer copies or define additional vsets that are accessible to the current user.

ARC-01179E List not allowed in /COPIES.

Cause: The COPIES qualifier does not accept a list of parameters. Only one parameter can be specified.

Action: Remove the list of parameters from the COPIES qualifier.

Example: ARCHIVE/DBSPACESET=(*)/COPIES=(1,2)

ARC-01182E No more than one transit can be specified.

Cause: Multiple transit vsets are specified in a single command.

Action: Remove all but one of the specified transit vsets.

ARC-01185E Request %s has already been taken care of.

Cause: The specified request is presently locked by another user. This occurs when the request is modified or executed.

Action: Wait until the lock has been released.

ARC-01186E One of the requests specified in a wait-event is missing.

Cause: One of the requests specified in the wait-event list does not exist or has been cancelled.

Action: Either cancel the current request or modify its wait-event list so that it no longer refers to the missing request.

ARC-01187E Request %s has been cancelled.

Cause: The attempted operation could not be completed because the specified request has been cancelled.

Action: Reenter the request.

ARC-01188E None of the required vsets are currently available.

Cause: None of the required vsets are currently available. This can occur if all the vsets are locked, if the available vsets are all full, or if the required vsets are not ONSITE.

Action: If all the vsets are full, define an additional volume. If all the vsets are locked, wait until they are unlocked. If the vsets are not ONSITE, recall one of them. If your operation involves the use of a transit vset, make sure to check it as well.

ARC-01189E No transit available for the network request.

Cause: No transit volume-sets are available for use by the request. This usually occurs when all the transit vsets are locked by other users or when they are all full.

Action: Verify that the transit vsets are not full. If they are not, wait until one of them is no longer locked.

ARC-01191E Need a transit vset for a network request.

Cause: The attempted operation could not be completed because it involved a network operation and an available transit vset could not be found.

Action: Verify that a transit vset exists and that is it accessible.

ARC-01193E Unable to define a new volume in the catalog.

Cause: An attempt to define a new volume failed.

Action: See the other error messages to determine why the volume definition failed.

ARC-01194E Invalid status %s.

Cause: The catalog records the status of the specified request as not being one of the supported types.

Action: Contact your Informix Technical Support representative.

ARC-01195E I/O error while verifying the save set.

Cause: While verifying the save set, an I/O error occurred. This error is usually caused by a timing problem or by a defective device or volume.

Action: Try using a different device. If the error persists, use another copy.

ARC-01196E I/O error while verifying file.

Cause: While verifying a file, an I/O error occurred. This error is usually caused by a timing problem or by a defective device or volume.

Action: Try using a different device. If the error persists, try using a different volume.

ARC-01196E I/O error while verifying file.

Cause: While verifying the save set, an I/O error occurred. This error is usually caused by a timing problem or by defective media.

Action: Rearchive the data. If the error persists, use other media.

ARC-01198E I/O error while writing Save-set.

Cause: While writing the save set, an I/O error occurred. This error is usually caused by a timing problem or by a defective device. This error might also occur when using third-party peripherals that are not fully compatible with your computer hardware/operating-system specifications.

Action: Re-archive the data. If the error persists, use another medium. If you are using third-party peripherals, call your Informix sales representative and verify that the device has been certified for use.

ARC-01199E Request %s has been cancelled during execution.

Cause: The specified request has been cancelled during execution either by the user or by an operator.

Action: Reenter the request using MODIFY/COMMAND if you want the request to be executed.

ARC-01201E I/O error while reading file.

Cause: While reading from the save set, an I/O error occurred. This is usually caused by a timing problem or by a defective device or volume.

Action: Reread the data. Try a different device or a copy of the volume. If the error persists, the media are defective and can no longer be used.

ARC-01202E Communication to cataloger not open.

Cause: ON-Archive attempted to send a message to the cataloger but was unsuccessful.

Action: Check that **oncatlgr** is running and that s a pipe file in with the name **IX_CAT_6.0.*servernum*** is in **/tmp**.

ARC-01203E No more communication ports available.

Cause: There are more than 99 ON-Archive users.

Action: No more ON-Archive users can start until some of the existing users exit.

ARC-01206E Status of information to send is invalid.

Cause: ON-Archive is attempting to send a message before a response from the previous message has been received.

Action: ON-Archive is confused; stop the process and restart it.

ARC-01207E Invalid communication port.

Cause: The cataloger received a message from an **onarchive** process it does not recognize.

Action: Stop and restart **oncatlgr** and **onarchive**.

ARC-01208E Unable to send a message to the cataloger.

Cause: ON-Archive tried to send a message to **oncatlgr**, but **oncatlgr** did not respond.

Action: Check that **oncatlgr** is up and running.

ARC-01209E Unable to read a message from the cataloger.

Cause: ON-Archive tried to receive a message from **oncatlgr** but was timed out before it received a response from **oncatlgr**.

Action: Check that **oncatlgr** is up and running.

ARC-01211E The cataloger is not installed.

Cause: The attempted operation could not be completed because the ON-Archive cataloger is not currently installed.

Action: Install the cataloger using the **start_oncatlgr** command.

ARC-01216E Unable to send the ready-to-read flag.

Cause: IPC send through pipe failed.

Action: Restart **oncatlgr** and **onarchive**.

ARC-01217F Error message received from cataloger; see cataloger output.

Cause: An internal error has occurred that requires immediate attention by the ON-Archive administrator.

Action: Report the problem to your ON-Archive administrator. You can find the details of the error in the file **/tmp/oncatlgr.*pid***.

ARC-01219E Unable to cancel a specific request.

Cause: The cataloger received a request from an **onarchive** process it does not recognize.

Action: Check that not more than one **oncatlgr** process is running for this database server. Restart **oncatlgr** and **onarchive**.

ARC-01221I Undefined user default file.

Cause: Neither the user-specified user default file nor the installation default user default file (**$INFORMIXDIR/etc/oper_deflt.arc**) could be found.

Action: Check that either the user-specified user default file or **$INFORMIXDIR/etc/oper_deflt.arc** exists and is accessible.

ARC-01225I Volume is currently locked by another user.

Cause: The attempt to use the specified volume failed because it is currently locked by another user.

Action: Wait until the volume is no longer locked.

ARC-01226I Vset is currently locked by another user.

Cause: The attempt to use the specified vset failed because it is currently locked by another user.

Action: Wait until the vset is no longer locked.

ARC-01227I Vset is not empty.

Cause: The DELETE command specified refers to a vset that is not empty. At least one of its volumes contains data.

Action: Remove the data from all of the volumes, then delete the vset.

ARC-01229W User(s) not added to "SYSTEM" vset.

Cause: By definition, all users have access to a vset defined as SYSTEM class. Specific users cannot be restricted from using or granted access to use a vset that is defined as SYSTEM class.

Action: Modify the class of the vset to USER, then specify which users have access to it or do not use the USER qualifier.

ARC-01230E Cannot start request - predecessors not completed.

Cause: An EXECUTE command was entered for a request with a wait event tied to another request that had not yet completed.

Action: Wait until the required events have completed or MODIFY the wait event list of the request.

ARC-01232W Insufficient access privilege to vset.

Cause: The executing user does not have operating-system permission to access the vset on disk.

Action: Verify that the vset directory exists, that the user has permission to access it, and that the user has permission to access the **VOL.HDR** and ***.SAV** files.

ARC-01234W /PERMANENTLY_MOUNTED reset to /NOPERMANENTLY_MOUNTED.

Cause: A user specified a tape to be PERMANENTLY_MOUNTED, which is not allowed by ON-Archive.

Action: No action needed; ON-Archive reset the value for you.

ARC-1239W The archiving may be incomplete!

Cause: An error occurred trying to close out a volume for the archive, logical log backup, or copy operation.

Action: Retry the request with a different volume.

ARC-01241E Invalid communication port.

Cause: The cataloger received a message from an **onarchive** process it does not recognize.

Action: Stop and restart **oncatlgr** and **onarchive**.

ARC-01242E Error trying to send an answer.

Cause: Either **oncatlgr** or **onarchive** tried to establish and send information over their pipe connection but failed.

Action: Verify that the pipe file is located in **/tmp** and that **oncatlgr** is running.

ARC-01243E	Error trying to complete an answer.
Cause:	Either **oncatlgr** or **onarchive** tried to establish and send information over their pipe connection but failed.
Action:	Verify that the pipe file is located in **/tmp** and that **oncatlgr** is running.
ARC-01244E	Error occurred while reading a message.
Cause:	Either **oncatlgr** or **onarchive** tried to read information over their pipe connection but failed.
Action:	Verify that the pipe file is located in **/tmp** and that **oncatlgr** is running.
ARC-01245E	Error while trying to close a communication port.
Cause:	Either **oncatlgr** or **onarchive** attempted to close their pipe communications and got an error.
Action:	None needed.
ARC-01250E	Error trying to test if the catalog is already installed.
Cause:	Another process is already using the pipe file.
Action:	Check that only one **oncatlgr** process exists per database server.
ARC-01254E	Error trying to add a logon request.
Cause:	The **oncatlgr** process got an error trying to add another **onarchive** process to the list of processes with which it is communicating.
Action:	Too many **onarchive** processes might already be logged on to **oncatlgr**. Wait until some of them finish.
ARC-01255E	Error adding a request.
Cause:	The **oncatlgr** process got an error trying to add another **onarchive** process to the list of processes with which it is communicating.
Action:	Too many **onarchive** processes might already be] logged on to **oncatlgr**. Wait until some of them finish.

ARC-01256E Request received from a PID that is not logged in the cataloger.

Cause: The **oncatlgr** process received a message from an **onarchive** it does not recognize.

Action: Stop and restart **oncatlgr** and **onarchive**.

ARC-01257E Unable to put a request in the ready-to-read mode.

Cause: The **oncatlgr** process received a message from an **onarchive** that is in a state it does not recognize.

Action: Stop and restart **oncatlgr** and **onarchive**.

ARC-01258E Error trying to send an error message.

Cause: The **oncatlgr** process got an error trying to communicate an error message to an **onarchive** process.

Action: Stop and restart **oncatlgr** and **onarchive**.

ARC-01261E Error trying to send a cancel message.

Cause: The **oncatlgr** process got an error trying to communicate a cancel message to an **onarchive** process.

Action: If the **onarchive** process is running, stop it and then restart it.

ARC-01262E Error reading a message from a cancel call.

Cause: The **oncatlgr** process did not recognize the response it received after sending a cancel message to an **onarchive** process.

Action: If the **onarchive** process is running, stop it and then restart it.

ARC-01263E Error while adding a vset user.

Cause: An error occurred while adding a vset user.

Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01264E Cannot allocate device %s.

Cause: An error occurred while allocating the specified device.
Action: Verify that the device specified exists and that it is on-line.

ARC-01265E Cannot deallocate device %s.

Cause: An error occurred while deallocating the specified device.
Action: Verify that the specified device is on-line.

ARC-01266E Cannot dismount device %s.

Cause: An error occurred while dismounting the specified device.
Action: Verify that the specified device is on-line.

ARC-01267E Tape not initialized.

Cause: An error occurred while trying to initialize (write a volume label) the tape.
Action: Reexecute the command. If the error persists, use another medium.

ARC-01268E Disk not initialized.

Cause: An error occurred while trying to initialize (write a volume label) the directory specified on the disk drive.
Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01269E Tape not defined.

Cause: The specified tape was not defined for use by ON-Archive.
Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01270E Disk not defined.

Cause: The specified disk was not defined for use by ON-Archive.
Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01271E Cannot unlock volume %s.

Cause: The **oncatlgr** process might have received an error while trying to unlock the volume. The volume might still be marked as locked.

Action: Resolve any **oncatlgr** errors. Then, if the volume is marked as locked, run the **arc_purge.sql** script to unlock it.

ARC-01272E Cannot define vset %s.

Cause: Something prevented the vset from being defined.

Action: Check the other error messages to determine why the vset could not be defined.

ARC-01273E Vset not deleted.

Cause: An error occurred while trying to delete the specified vset.

Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01275E User not deleted from vset.

Cause: An error occurred while trying to delete a user from the specified vset.

Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01276E There is nothing to modify.

Cause: The MODIFY command did not result in any modifications to the catalog.

Action: Re-enter the MODIFY command so that at least one of the specified parameters or qualifiers is different from the current values in the ON-Archive catalog.

ARC-01278E Cannot get volume from catalog.

Cause: Something prevented **onarchive** from getting information about the requested volume from the catalog. The volume might not have been defined.

Action: Check any other error messages to determine why the volume could not be located in the catalog. Also check for any **oncatlgr** error messages.

ARC-01279E Cannot get vset from catalog.

Cause: The information from the specified vset cannot be retrieved from the catalog.

Action: Verify the spelling of the specified vset.

ARC-01281E Vset currently locked by another user.

Cause: The attempt to use the specified vset failed because it is currently locked by another user.

Action: Wait until the vset is no longer locked.

ARC-01282E Volume currently locked by another user.

Cause: The attempt to use the specified volume failed because it is currently locked by another user.

Action: Wait until the volume is no longer locked.

ARC-01284E Vset not modified.

Cause: The operation against the specified vset could not be completed; therefore, it was not modified.

Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01285E Request not modified.

Cause: The operation against the specified request could not be completed; therefore, it was not modified.

Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01286E File not modified.

Cause: The operation against the specified file could not be completed, therefore it was not modified.

Action: Refer to the other error messages which accompanied this message for more details as to why the error occurred.

ARC-01291E Cannot add the request in the catalog.

Cause: An error occurred while trying to record a request in the catalog. This error can be caused by several factors; for example, no more space exists in the database, the database crashed during the insertion of the row, or the catalog has been deleted.

Action: Verify that your database is functioning and that you can add data. Also verify that the tables that compose the catalog still exist.

ARC-01293E Error in /WAIT parameter(s).

Cause: There is an error in the WAIT parameter.

Action: Modify the parameter so that it conforms to the proper ON-Archive syntax. Refer to the reference material for the command or to the on-line Help facility for more information.

Example: ARCHIVE/DBSPACESET=*/WAIT=(R:147,148)

ARC-01294E Predecessor has been cancelled.

Cause: One of the requests specified in the wait-event list has been cancelled.

Action: Remove the cancelled wait-event from the request.

ARC-01296F Format file not found.

Cause: The message file that contains the informational messages could not be found. The file is called **fmt_E.dat** (unless another name was specified in the ON-Archive configuration file, **config.arc**) and is located in **$INFORMIXDIR/msg**.

Action: Check that the format message file exists and is accessible.

ARC-01297I Request %s not cancelled.

Cause: The specified request was not cancelled.

Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01298E Request status update failure.

Cause: An attempt to update the status of a request failed. The status might be in an incorrect state.

Action: Check other error messages to see why the attempt might have failed. Also check **oncatlgr** output for error messages.

ARC-01299E The unlock of the request failed.

Cause: An attempt to unlock the request in the archive catalog failed.

Action: Check other error messages to see why the attempt might have failed.

ARC-01302E Save-set read failed.

Cause: An attempt to read a save set failed.

Action: Check other messages to see why the read failed.

ARC-01304E The save-set is corrupted.

Cause: An inconsistency was detected during a save set read after the volume had been verified.

Action: Check other error messages to see why the read failed. If necessary, use another copy of the save set.

ARC-01307E Unix signal received. Signal number = %d.

Cause: UNIX received a signal. Typically, this causes **onarchive** to terminate.

Action: Determine from the signal number if this requires assistance from Informix technical support.

ARC-01310E Cannot modify the request status.

Cause: An attempt to modify the status of a request failed. The status might be in an incorrect state.

Action: Check other error messages to see why the attempt might have failed. Also check **oncatlgr** output for error messages.

ARC-01313E Retrieve request failed.

Cause: The attempted retrieve request failed.

Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01315E List not allowed in database.

Cause: The qualifier DATABASE does not accept a list of parameters. Only one parameter can be specified.

Action: Enter a single database name or *, which lists all the databases.

ARC-01317E Cannot get request information.

Cause: A command was entered that specified a non-existent request id, or that resulted in no request ids matching the qualifier list specified in the command.

Action: Modify the specified request id (or its qualifier list) to one that exists, or use the wildcard character (*) to specify all request ids.

ARC-01322E Error reading ANSI volume label.

Cause: A problem occurred while trying to write the ANSI volume label on an archival medium (tape or disk). This is usually caused by a timing problem or by defective media. This might also occur when using third-party peripherals that are not fully compatible with the hardware / operating-system specifications of your computer.

Action: Verify that the tape drive is on-line and that the proper tape is mounted.

ARC-01323E Error writing the ANSI volume label.

Cause: A problem occurred while trying to write the ANSI volume label on an archival medium (tape or disk). This error is usually caused by a timing problem or by defective media. This error might also occur when you use third-party peripherals that are not fully compatible with the hardware/operating-system specifications of your computer.

Action: Verify that the drive is on-line and is not write disabled. If you are using third-party peripherals, call your Informix sales representative and verify that the device has been certified for use.

ARC-01324E Error writing the ANSI HDR1.

Cause: A problem occurred while trying to write the ANSI volume label on an archival medium (tape or disk). This error is usually caused by a timing problem or by defective media. This error might also occur when you use third-party peripherals that are not fully compatible with the hardware/operating-system specifications of your computer.

Action: Verify that the drive is on-line and is not write disabled. If you are using third-party peripherals, call your Informix sales representative and verify that the device has been certified for use.

ARC-01325E Error writing the ANSI HDR2.

Cause: A problem occurred while trying to write the ANSI volume label on an archival medium (tape or disk). This error is usually caused by a timing problem or by defective media. This error might also occur when you use third-party peripherals that are not fully compatible with the hardware/operating-system specifications of your computer.

Action: Verify that the drive is on-line and is not write disabled. If you are using third-party peripherals, call your Informix sales representative and verify that the device has been certified for use.

ARC-01326E Error writing the ANSI HDR3.

Cause: A problem occurred while trying to write the ANSI volume label on an archival medium (tape or disk). This error is usually caused by a timing problem or by defective media. This error might also occur when you use third-party peripherals that are not fully compatible with the hardware/operating-system specifications of your computer.

Action: Verify that the drive is on-line and is not write disabled. If you are using third-party peripherals, call your Informix sales representative and verify that the device has been certified for use.

ARC-01327E Error writing the ANSI HDR4.

Cause: A problem occurred while trying to write the ANSI volume label on an archival medium (tape or disk). This error is usually caused by a timing problem or by defective media. This error might also occur when you use third-party peripherals that are not fully compatible with the hardware/operating-system specifications of your computer.

Action: Verify that the drive is on-line and is not write disabled. If you are using third-party peripherals, call your Informix sales representative and verify that the device has been certified for use.

ARC-01332E Error while adding file %d to catalog.

Cause: An error occurred while trying to record an archive or backup file in the catalog. This error can be caused by several factors; for example, the database has no more space, the database crashed during the insertion of the row, or the catalog has been deleted.

Action: Verify that your database is functioning and that you can add data. Also verify that the tables that compose the catalog still exist.

ARC-01341E Error while reading save-set HDR1.

Cause: An I/O error occurred while ON-Archive was trying to read a save set from a tape device. The actual cause is noted by the operating-system error that is printed out as a companion to this error.

Action: Determine why the operating system does not permit the I/O to occur. The tape might be bad.

ARC-01342E Save-set file not found.

Cause: The specified save set was not found on the volume.

Action: Use the list commands to find out on which vset and volume the save set was placed. For disk vsets, make sure that the save set file is still in the volume directory.

ARC-01343E Error while deleting the save-set.

Cause: An error occurred during the deletion of a save set that was archived on a DISK save set.

Action: Verify that the save set exists in the expected directory, and that the save set is not locked by another user.

ARC-01355E Comment too long, maximum length is 80.

Cause: The comment entered is longer than the maximum length supported.

Action: Shorten the length of the comment.

ARC-01356E Parameters too long, maximum length is 20.

Cause: The maximum length of the PARAMETER qualifier is 20 characters.

Action: Modify the PARAMETER qualifier so that it is not more than 20 characters.

ARC-01357E Virtual too long, maximum length is 240.

Cause: The parameter of the VIRTUAL qualifier is longer than the maximum length supported: 255 characters.

Action: Modify the VIRTUAL qualifier so that it is not more than 255 characters. You can use logical names to reduce the size of the qualifier.

ARC-01358E Invalid max space.

Cause: An invalid maximum space was specified during the definition of a volume.

Action: Specify a valid maximum space for the volume.

ARC-01359E Cannot delete file %s.

Cause: An error occurred while attempting to delete the specified file.

Action: Verify that the device and file specification of the file is correct.

ARC-01360E Max space is too large, range is 16 to 99999999.

Cause: The maximum space specified is too large.

Action: Modify the maximum space specified so that it is in the range of sizes supported.

ARC-01361E Missing username after /USER.

Cause: The USER qualifier requires a username parameter when used in its current context.

Action: Supply the missing username.

ARC-01366E Request currently locked by another user.

Cause: The operation against the specified request could not be completed because the request is locked by another user. For example, two users cannot attempt to retrieve information from the same request at the same time.

Action: Wait until the request is no longer locked.

ARC-01367E	There is no password in catalog for the current request.
Cause:	A request has been specified with DECRYPT, but the original request has no UEMK. The original request probably was not encrypted.
Action:	Do not specify DECRYPT.
ARC-01368E	Missing /VIRTUAL qualifier for a DISK vset.
Cause:	The VIRTUAL qualifier is mandatory for the definition of a disk vset.
Action:	Supply the missing VIRTUAL qualifier in the command.
Example:	DEFINE/VSET=TEST/VOLUME/MAX_SPACE=99999
ARC-01371E	Request is not new anymore.
Cause:	The specified request does not have a status of NEW; therefore, the operation cannot be performed.
Action:	Reenter the command, so that the request has a status of NEW.
ARC-01373E	Request id list not permitted, single value only.
Cause:	The command entered does not support a list of request ids. Only one request id can be specified.
Action:	Specify only a single request id.
ARC-01375W	No vset found.
Cause:	A command specified a nonexistent vset.
Action:	Verify the spelling of the vset. Specify the correct vset or use the wildcard character (*) to specify all vsets.

ARC-01377W No file found.

Cause: A command specified a non-existent file or a LIST/FILE command was entered that did not return any information based on the qualifiers specified.

Action: Change the file specification or use the wildcard character (*) to specify all files. You might also change some of the qualifiers that filter the files that can be returned from the query of the catalog. The most common error made is to forget to include the /USER qualifier. If you wish to list information pertaining to ALL users, you must add the qualifier /USER=* to your command (assuming that this is not already specified in your default file).

ARC-01379W No request available.

Cause: No requests were found that matched the qualifier list of the command specified.

Action: Modify the qualifier list. The most common error is to forget to include the USER qualifier. If you wish to list information pertaining to ALL users, you must add the qualifier USER=* to your command (assuming that this is not already specified in your default file).

ARC-01381E Missing /USER parameter

Cause: The command specifies /CLASS=USER, which requires the /USER qualifier and a list of users. The /USER qualifier, or parameter, is missing.

Action: Modify the command to add the /USER qualifier and a list of users, or change the class to /SYSTEM.

ARC-01382E Request number is not in the current range.

Cause: From within the operator interface of the EXECUTE command, the request number selected is not in the current range of request numbers displayed. (The left-most column lists the request numbers, not to be confused with the request id, which is found in the right-most column)

Action: Press RETURN one or more times until the specified request number is displayed.

ARC-01383E Archiving of the file(s) is not yet completed.

Cause: An attempt to cancel or remove data from a request whose status is PARENT was made. PARENT requests cannot be cancelled nor can their data be removed from the catalog until all of their child requests have been completed or cancelled.

Action: Execute and / or cancel the child requests of the parent and then reexecute the command.

ARC-01392I Volume %s not deleted.

Cause: The attempted deletion of the specified volume was not successful.

Action: Refer to the other error messages that accompanied this message for more details as to why the error occurred.

ARC-01393E Volume %s protected against deletion.

Cause: The volume specified for deletion could not be deleted because it does not have 'delete' (D) access.

Action: Modify the access protection on the volume so that deletion is permitted. Use the MODIFY / VOLUME command.

ARC-01395E At least one volume is delete protected.

Cause: One (or more) of the specified volumes cannot be deleted because it does not have 'delete' (D) access.

Action: Modify the access protection of the affected volumes so that deletion is permitted. Use the MODIFY/VOLUME command.

ARC-01399E Request is not encrypted.

Cause: The **onkeymgr** utility was asked to decrypt the UEMK for a request but found no UEMK for the request.

Action: None needed.

ARC-01400E Error while encrypting password.

Cause: The password that was supplied could not be encrypted using the specified encryption method.

Action: Supply a different password that meets the requirements of the specified encryption method.

ARC-01404E Unable to catalog: foreign vset

Cause: The CATALOG command cannot be used against a vset that is defined as /FOREIGN.

Action: Select another vset to catalog.

ARC-01405E Unable to catalog: vset is not ONSITE

Cause: The CATALOG command cannot be used against a vset that is OFFSITE.

Action: Modify the location of the vset to ONSITE.

ARC-01414E Invalid save-set data type.

Cause: The save set accessed does not have the correct internal data type.

Action: This save set cannot be read using ON-Archive.

ARC-01415E List not allowed in /VSET.

Cause: The VSET qualifier does not accept a list of parameters. Only one parameter can be specified.

Action: Remove the list of parameters from the VSET qualifier.

Example: COPY/DESTINATION=NEW/VSET=(TEST1,TEST2)

ARC-01416E List not allowed in /DESTINATION.

Cause: The DESTINATION qualifier does not accept a list of parameters. Only one parameter can be specified.

Action: Remove the list of parameters from the DESTINATION qualifier.

Example: COPY/VSET=TEST1/DESTINATION=(TEST1,TEST2)

ARC-01417E Cannot copy on a transit vset.

Cause: The destination vset of a copy request cannot be a transit vset.

Action: Specify a regular vset as the final destination.

ARC-01418E Volume-set %s is imported.

Cause: A command was attempted on a vset that was defined as IMPORTED, which is not permitted on this type of vset. For example, archiving is not permitted on a imported vset.

Action: Specify another vset that is not defined as IMPORTED for this command.

ARC-01419E Volume-set %s is foreign.

Cause: A command was attempted on a vset that was defined as FOREIGN. The command is not permitted on this type of vset. For example, archiving is not permitted on a foreign vset.

Action: Use a vset that is not defined as FOREIGN.

ARC-01420E The destination vset can not be the same as the source vset.

Cause: The specified command attempted to copy a vset onto itself.

Action: Change either the source vset or the destination vset so that they are not identical.

Example: COPY/VSET=TEST/DESTINATION=TEST

ARC-01421E Request %s is cancelled or failed.

Cause: The operation on the specified request could not be completed because the status of the request is either COMPLETED or UNSUCCESSFUL.

Action: Specify another request for the operation.

ARC-01422E User must be an operator to copy an entire vset.

Cause: A user attempted to copy an entire vset containing save sets belonging to other users, without having operator privileges. Only operators have access to every user's save sets. Regular users do not have enough privileges to copy save sets belonging to other users.

Action: Have an ON-Archive operator execute the COPY command.

ARC-01423E Volume-set required by a request.

Cause: An attempted DELETE/VSET command could not be completed because one or more pending requests require the specified vset.

Action: Modify the pending request so that it no longer requires the use of the specified vset.

ARC-01425E The volume %s is not empty.

Cause: An attempt was made to delete a volume that contained one or more save sets. Before you can delete a volume from the ON-Archive catalog, you must first remove all the save sets from it.

Action: Remove the remaining save sets using the REMOVE / REQUEST command.

ARC-01426E Invalid system ID or invalid cluster ID.

Cause: ON-Archive was installed on a computer for which it was not licensed.

Action: Install ON-Archive on the computer for which it was licensed, or call your local sales representative to discuss upgrading your license for a new computer.

ARC-01429E Error in default file.

Cause: The first noncomment line in the **oper_deflt.arc** file must be the word ENGLISH. This was not found.

Action: Check that the **oper_deflt.arc** file (or the file specified in the ON-Archive configuration file, **config.arc**, for the default file) has the proper format.

ARC-01430E No input vset name found after / VSET.

Cause: A command that included the VSET qualifier did not include a vset name.

Action: Supply the missing vset name in the command.

ARC-01431E No vset name found after / DESTINATION.

Cause: A command that included the DESTINATION qualifier did not include a vset name.

Action: Supply the missing vset name in the command.

ARC-01435E The specified request id does not exist.

Cause: The specified request id does not exist in the catalog. It is possible that the request id previously existed in the catalog, but it has since been removed.

Action: If the save set still resides on your physical archival medium, you can catalog the media and the request will be regenerated in the on-line catalog.

ARC-01436E Invalid copy comment.

Cause: The comment field of the child-copy request has been corrupted. The comment field contains information such as the parent request-id and the number of copies.

Action: Cancel the child-copy request and either reenter the original parent request or manually enter a copy of the parent request.

ARC-01437E Wrong parent status.

Cause: The status of the parent request of the current request is not PARENT. The two requests are not properly synchronized in the catalog.

Action: If the child request was for a COPY, manually enter a COPY request using the parent request as the source.

ARC-01438E The vset is not available at this time.

Cause: The specified vset is not available because it is locked by another user, or it is not ONSITE.

Action: Specify another vset or wait until the volume is no longer locked or it is ONSITE.

ARC-01440E The request is already on the output vset.

Cause: The output vset specified already contains a copy of the data.

Action: Specify another output vset as the destination of the request. For security reasons, ON-Archive does not allow two copies of the same data to reside on the same vset.

ARC-01442I	No available volume found.
Cause:	The attempted operation could not be completed because the destination volume is not available. This error usually occurs either because the vset is locked by another user or the vset is full.
Action:	Verify that the vset is not locked by another user, and that it is not full.
ARC-01443I	Request %s cancelled.
Cause:	The specified request was cancelled.
Action:	Reenter the request using MODIFY / COMMAND if you want the request to be executed.
ARC-01447E	Missing / MAX_SPACE qualifier for a DISK vset.
Cause:	The DEFINE / VOLUME command for a DISK vset is missing the mandatory qualifier MAX_SPACE.
Action:	Supply the missing MAX_SPACE qualifier. Refer to the reference material for the command or to the on-line Help facility for more information.
ARC-01450E	Either database %s is undefined or it has no tables defined.
Cause:	The LIST / DATABASE command was invoked on a database that is unknown to the database server.
Action:	Check the name of the database and resubmit with a valid database name.

ARC-01456E Unable to add save-set in catalog.

Cause: An error occurred while trying to record the save set in the catalog. This error can be caused by several factors; for example, the database has no more space, the database crashed, or the catalog has been deleted.

Action: Verify that your database is functioning and that you can add data. Verify that the tables that compose the catalog still exist and then rearchive the data.

Verify that each line in the DXS file that begins with a (.) is followed immediately by a valid ARCHIVE*SQL keyword. Each keyword, except for RETRIEVE, must be followed by a minimum of one blank space and the corresponding parameter.

ARC-01470E Default line too long in default file.

Cause: One of the lines in the default file, **oper_deflt.arc**, is longer then the maximum length supported.

Action: Shorten the line in the default file.

ARC-01483I No data to archive.

Cause: The specified BACKUP operation did not result in any data being archived.

Action: This error can occur when there are no unbacked-up log files.

ARC-01484E Too many parameters in /SIZE qualifier.

Cause: The SIZE qualifier contains more than two parameters.

Action: Remove the extra parameters from the SIZE qualifier list. The SIZE qualifier can only have one or two parameters.

Example: ARCHIVE/FILE=(*)/SIZE=(1,2,3)

ARC-01487E Undefined save-set.

Cause: The specified save set is not in the ON-Archive catalog.

Action: If the save set is on a volume, it might have been deleted from the catalog. You can reenter it into the catalog using the CATALOG command.

ARC-01499E /VOLUME should not have parameter in this context.

Cause: The specified command does not support a parameter for the VOLUME qualifier.

Action: Remove the unnecessary parameter.

Example: DEFINE/VSET=TEST/VOL=NEWVOL

ARC-01501E /MAX_SPACE cannot be changed when volume is not empty.

Cause: The maximum space that a DISK-based volume is allowed to occupy (specified by the MAX_SPACE qualifier) cannot be modified unless the volume is empty.

Action: Empty the volume in question by either copying and removing the save sets that are stored in the volume or by just removing them.

ARC-01504E The destination vset is not available.

Cause: The attempted operation could not be completed because the destination vset is not available. This error usually occurs either because the vset is locked by another user, all the volumes of the vset are full, or the location of the vset is not ONSITE.

Action: Verify that the vset is not locked by another user, that all its volumes are not full, and that its location is ONSITE.

ARC-01505E List not allowed in /OUTPUT.

Cause: The OUTPUT qualifier does not accept a list of parameters. Only one parameter can be specified (but it can include any supported wildcard characters).

Action: Remove the list of parameters from the OUTPUT qualifier.

Example: RETRIEVE/DBSPACESET=(*)/REQUEST=
1/OUTPUT=(A.A,B.B,C.C)

ARC-01506E Missing device name in virtual specification.

Cause: The specification of the virtual directory does not include the device name.

Action: Supply the device name in the virtual directory specification.

Example:
```
DEFINE/VSET=TEST/VOL-/MAX_SPACE=10000-
/VIRTUAL=(/directory)
```

ARC-01507E Number of copies is out of range (1 - 5).

Cause: The number of copies specified (using the COPIES qualifier) is not in the range of 1 to 5.

Action: Modify the number of copies requested to be in the range of 1 to 5.

Example:
```
ARCHIVE/DBSPACE=(*)/COPIES=10
```

ARC-01509E Request not resubmitted.

Cause: An error was detected while ON-Archive was trying to resubmit a request automatically because of the /REPEAT qualifier in the request.

Action: Examine the request to determine the problem. This message is usually accompanied by another message that describes the nature of the problem.

ARC-01510E List not allowed in /REPEAT.

Cause: The REPEAT qualifier does not accept a list of parameters. Only one parameter can be specified.

Action: Remove the list of parameters from the REPEAT qualifier.

Example:
```
BACKUP/LOGFILE/AUTOMATIC/
REPEAT=(1-00:00:00,2-00:00:00)
```

ARC-01516E	Invalid Request Type for Retrieve Operation.
Cause:	The original request id specified in the RETRIEVE command is not of the correct type to be retrieved. For example, if the original request archived some database data and then a RETRIEVE / FILE command was entered against the original request, you would receive this message because the type of data to be retrieved is not the same as that archived.
Action:	Use the LIST / REQUEST command to view the type of the original request.
ARC-01517E	Invalid Request Status for Retrieve Operation.
Cause:	The status of the original request id specified in the RETRIEVE command does not permit data to be retrieved. For example, you cannot RETRIEVE data from a request whose status is NEW (not yet archived).
Action:	Verify the status of the original request.
ARC-01519E	The request is not on this volume-set.
Cause:	The REMOVE command was specified with a vset that does not contain the specified element.
Action:	Use LIST commands to determine on which vset the save set is stored.
ARC-01521W	The request %s is already on the output vset.
Cause:	The output vset specified already contains a copy of the data.
Action:	Specify another output vset as the destination of the request. For security reasons, ON-Archive does not allow two copies of the same data to reside on the same vset.
ARC-01522E	Volume(s) read protected.
Cause:	The volume specified could not be read because it does not have 'read' (R) access.
Action:	Modify the access protection on the volume, using the MODIFY / VOLUME command.

ARC-01524E Missing /SID qualifier for imported volume-set.

Cause: The command specified against an imported vset requires the SID qualifier (CPU System Identifier).

Action: Include the SID qualifier in the command. If you are unsure of the meaning of the SID qualifier, refer to the reference material for the CATALOG command or to the on-line Help facility.

ARC-01525E Invalid SID.

Cause: The SID (CPU system identifier) qualifier was entered with an invalid value.

Action: Enter a SID that is all digits and is greater than 0 and less than or equal to 999.

ARC-01526E /SID not permitted for local volume-set (not imported).

Cause: When cataloging a local vset (not imported), the SID (CPU system identifier) qualifier was specified. SID can only be applied to imported vsets.

Action: Remove the SID qualifier from the CATALOG command.

ARC-01529E Cannot modify a pending child request.

Cause: You attempted to modify one of the qualifiers of a pending archive child request. Archive child requests must be exact duplicates (except for the destination vset) of their parents.

Action: Cancel the pending archive child request as well as its parent request and then reenter the request with the correct qualifiers.

ARC-01530E Error during compression of file(s).

Cause: **onarchive** detected an error when trying to compress a file.

Action: If you provided your own compression routines, debug the routines. Otherwise, refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01531E Error during decompression of file(s).

Cause: **onarchive** detected an error when trying to compress a file.

Action: If you provided your own compression routines, debug the routines. Otherwise, refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01532E Error during encryption of file(s).

Cause: **onarchive** detected an error when trying to compress a file.

Action: If you provided your own compression routines, debug the routines. Otherwise, refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01533E Error during decryption of file(s).

Cause: **onarchive** detected an error when trying to compress a file.

Action: If you provided your own compression routines, debug the routines. Otherwise, refer to Appendix B, "Trapping Errors,", in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01534W Compress qualifier doesn't match that of the archived request.

Cause: The decompression method specified does not match the compression method used when the data was archived.

Action: Change the specified decompression method to match the method that was used during the original archive or backup operation.

ARC-01535W Encrypt qualifier doesn't match that of the archived request.

Cause: The decryption method specified does not match the encryption method used when the data was archived.

Action: Change the specified encryption method to match the method that was used during the original archive or backup operation.

ARC-01536I Retrieved data may be corrupted.

Cause: Executing the specified RETRIEVE command might corrupt the retrieved data. This error typically occurs if you attempt to retrieve data using a different decryption or decompression method than was used during the archive or backup operation.

Action: Specify the correct decryption and / or decompression methods as part of the RETRIEVE command.

ARC-01544E writev () error.

Cause: **onarchive** encountered an error when trying to communicate with **oncatlgr**. Most likely, the operation timed out.

Action: Verify that **oncatlgr** is alive and operational. Restart **onarchive**.

ARC-01557E open () error.

Cause: **onarchive** encountered an error when trying to communicate with **oncatlgr**. Most likely, the operation timed out.

Action: Verify that **oncatlgr** is alive and operational. Restart **onarchive**.

ARC-01558E readv () error.

Cause: **onarchive** encountered an error when trying to communicate with **oncatlgr**. Most likely, the operation timed out.

Action: Verify that **oncatlgr** is alive and operational. Restart **onarchive**.

ARC-01558I Tape selection has been cancelled.

Cause: During the allocation of a tape device, the operation was cancelled.

Action: Reexecute the operation if the cancellation request was made in error.

ARC-01560E Cannot skip tape mark.

Cause: ON-Archive was attempting to do a MTFSF **ioctl** call to skip one or more save sets on a tape and detected an error.

Action: Check the operating-system error number to see why the skip-file call failed.

ARC-01564E Cannot open tape.

Cause: An operating-system error was encountered when attempting to open a tape device. The operating-system error number is printed in a message following this one.

Action: Check the documentation for your operating system to determine the problem with the tape device. Possible problems are as follows: the permissions on the device file do not permit access by this user, the device is not connected properly, or the device is not configured to ON-Archive properly.

ARC-01563E Cannot parse file specification

Cause: The file specification does not follow the file-naming conventions defined by your operating system.

Action: Modify the file specification so that it conforms to the file-naming conventions of your operating system.

ARC-01565E Unknown SCSI generic tape drive.

Cause: The tape drive type connected to the SCSI port is unknown.

Action: ON-Archive does not currently support this type of tape drive. If you would like to obtain support for this drive, contact your local sales representative.

ARC-01566E The executor doesn't have read and search access on directory.

Cause: The attempted operation could not be completed because the executor does not have read and search access on the full directory path of the file specified.

Action: Modify the protection on the directory path so that the executor has both read and search access on the complete directory path of the file specified.

ARC-01567I The executor doesn't have read access on %s.

Cause: During execution of a request, ON-Archive attempted to archive a file to which the executor does not have read access. This error typically occurs because the file is locked by the operating system (usually when the file is being written to).

Action: Verify that you have the correct privileges to read the specified file.

ARC-01568E Parameter too long for %s.

Cause: The parameter entered for the specified qualifier is longer than the maximum length supported.

Action: Shorten the length of the parameter.

ARC-01569E List not allowed for /DRIVER.

Cause: The DRIVER qualifier does not accept a list of parameters. Only one parameter can be specified.

Action: Remove the list of parameters from the DRIVER qualifier.

Example: DEFINE/VSET=TEST/ACCESSIBILITY=50/CLASS=SYSTEM/DEVICE_TYPE=USR/DRIVER=(DISK,TAPE)

ARC-01570E List not allowed for /ACCESSIBILITY.

Cause: The ACCESSIBILITY qualifier does not accept a list of parameters. Only one parameter can be specified.

Action: Remove the list of parameters from the ACCESSIBILITY qualifier.

Example: DEFINE/VSET=TEST/CLASS=SYSTEM/DEVICE_TYPE=USR/DRIVER=DISK/ACCESSIBILITY=(50,60)

ARC-01571E Invalid protection, any of (RWD) allowed.

Cause: An invalid access protection type was specified.

Action: Specify only valid access protection types (Read, Write, or Delete).

ARC-01572I No more writable volume available on vset %s.

Cause: The remaining available volumes in the specified vset are not writable. The volumes in this vset are either not ONSITE, full, or do not have write-access protection.

Action: Verify why the volumes are not ONSITE and/or do not have write access protection. Either change one or more volumes to ONSITE, add write protection, or define a new volume for the vset.

ARC-01574E List not allowed in /MAX_SPACE.

Cause: The MAX_SPACE qualifier does not accept a list of parameters. Only one parameter can be specified.

Action: Remove the list of parameters from the MAX_SPACE qualifier.

Example: DEFINE/VOLUME/VSET=TEST/VIRTUAL=(/usr/test_vset/vol1)/MAX_SPACE=(10000,20000)

ARC-01575E List not allowed in /CLASS.

Cause: The CLASS qualifier does not accept a list of parameters. Only one parameter can be specified.

Action: Remove the list of parameters from the CLASS qualifier.

Example: DEFINE/VSET=TEST/ACCESSIBILITY=50/DRIVER=DISK/DEVICE_TYPE=USR/CLASS=(SYSTEM,USER)

ARC-01576E Unexpected internal fatal error has occurred.

Cause: The process encountered a fatal error. A core file is being produced for the process and a **/tmp/sysfail.***pid* file is being created, where *pid* is the process id of the failed ON-Archive process.

Action: Refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information. You will probably be asked to send them the core file.

ARC-01577I Program will be aborted to produce core dump.

Cause: An internal system error has been detected that will cause ON-Archive to abort the current operation.

Action: Consult the **/tmp/sysfail.***pid* file to see the details of the error, where *pid* is the process id of the failed ON-Archive process. Refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01578I Consult your **/tmp/sysfail.*%s*** file.

Cause: An internal ON-Archive error has occurred.

Action: Consult the **/tmp/sysfail.***pid* file to see the details of the error. Refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01579E List not allowed in /STATUS.

Cause: The STATUS qualifier does not accept a list of parameters. Only one parameter can be specified.

Action: Remove the list of parameters from the STATUS qualifier.

Example: LIST/REQ=*/STATUS=(NEW,PARENT)

ARC-01583E	Immediate database shutdown in progress, no operations are permitted
Cause:	The database server is in the process of shutting down. As a result, the ON-Archive cataloger either shuts down or goes into a hibernation state.
Action:	Stop and restart the cataloger using the **stop_catlgr** and **start_catlgr** commands after your database server has been restarted.
ARC-01584I	Initialization of tape cancelled at user's request.
Cause:	The user has decided not to initialize a tape after being prompted to do so.
Action:	No required action.
ARC-01585E	Host string syntax error.
Cause:	The remote-host-node name specified is unknown, or some other portion of the host string contains an error.
Action:	Verify that the host-node name specified is defined for your current node. Node names are defined in the hosts database if your network connections are made using TCP/IP or in the NCP file if your network connections are made using DECnet.
ARC-01587I	Internal check failure code: %s
Cause:	An internal consistency check failed and ON-Archive has determined that it cannot proceed.
Action:	Refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information. Also, report the code number that is returned in this message.
ARC-01588E	No volume label list permitted.
Cause:	More then one volume label was specified.
Action:	Reenter the current command, specifying only one volume label.

ARC-01589E Volume label cannot exceed 6 characters.

Cause: The volume label specified was longer than six characters.

Action: Reenter the volume label, specifying one that is six characters or less. This conforms with the ANSI standard for volume labels.

ARC-01590E The source vset is not available.

Cause: The attempt to use the specified vset failed because it is currently locked by another user.

Action: Wait until the vset is no longer locked.

ARC-01591E The remote virtual volume can be defined only on a network Vset.

Cause: An attempt was made to define a remote virtual volume on a vset that was not defined as a network vset.

Action: Define the remote virtual volume on a network vset. A network vset is a vset that has an owner node that is different from its device node.

ARC-01599I Archive request from %s can't be executed on this node.

Cause: The execution of the specified archive request requires a destination vset that is not local to the current node.

Action: Execute the request on a different node.

ARC-01600I Retrieve request from %s can't be executed on this node.

Cause: The execution of the specified retrieve request requires a destination vset that is not local to the current node.

Action: Execute the request on a different node.

ARC-01601I Can't copy on/from device_node on %s.

Cause: The execution of the specified request requires either a source vset or a destination vset that is not local to the current node, and that node has not been mounted by NFS.

Action: Execute the request on a different node or NFS mount the node defined for that vset.

ARC-01602I There is no NETWORK_MAIL entry in $ARC_CONFIG.

Cause: An attempt was made to send a mail message across the network using a protocol that was specified in the configuration file.

Action: Specify a different protocol for sending the network mail messages, or define the missing protocol in the configuration file. Protocols are specified in the configuration file by the keywords that begin with the characters 'NETWORK_MAIL_'.

ARC-01603I No mail has been sent to user %s.

Cause: No mail message was sent to the originator of the request. Refer to the other error messages that accompany this one for the details of why the mail message was not sent.

Action: Refer to the accompanying error messages for the required action.

ARC-01606E Invalid protocol for /PROTOCOL: %s.

Cause: The protocol specified by the /PROTOCOL qualifier is not supported in your release of ON-Archive.

Action: Contact your local ON-Archive sales representative to see if an upcoming release of ON-Archive will support the desired protocol.

ARC-01607I No /PROTOCOL was associated to the request.

Cause: The command specified requires the /PROTOCOL qualifier.

Action: Reenter the command adding the missing /PROTOCOL qualifier.

ARC-01610E Immediate Cataloger shutdown.

Cause: A user requested that the ON-Archive cataloger shut down by executing the **stop_catlgr** command. No further commands can be entered.

Action: Make sure that all users exit the ON-Archive system and then restart the cataloger using the **start_catlgr** command.

ARC-01720E Time out at the mount has been reached.

Cause: The time-out value for mount operations specified in the ON-Archive configuration file, **config.arc**, has been reached.

Action: Modify the value next to the TIME_OUT keyword in your ON-Archive configuration file, **config.arc**, and restart **oncatalgr**.

ARC-01721E Cannot define a volume for a remote Volume Set.

Cause: Defining a volume for a remote vset is not supported. A user must be directly connected to the same node as the node defined by the /DEVICE_NODE qualifier of the vset.

Action: Log onto the device node of the vset and reissue the DEFINE/VSET command.

ARC-01724I Input data can't be reached from this node, request %s still NEW.

Cause: The data specified to be archived cannot to accessed from the current node.

Action: Execute the request from another node from which the data is accessible.

ARC-01737I No failed request to purge.

Cause: The REMOVE/FAILED_REQUESTS command specified did not remove any requests from the catalog.

Action: This message is for information only. It is possible that there were no requests in the catalog to remove. If, however, you know that at least one request exists that has failed and should have been purged, specify a different set of qualifiers for the command. These qualifiers should be specified so that they do not eliminate the requests you wish to remove from the catalog.

ARC-01740E Invalid execution PHASE %s.

Cause: The execution phase specified for the command EXECUTE/PHASE is not supported.

Action: Specify a phase that is supported.

Example: EXECUTE/PHASE=unknown_phase

ARC-01762E Undefined vset %s.

Cause: The vset specified in the command is not defined in the ON-Archive catalog. It is also possible you do not have access to the specified vset.

Action: Use the LIST commands to list all the vsets known to the archive catalog to which you have access.

ARC-01769E /LOGFILE cannot have a value specified for it in this context.

Cause: The LOGFILE qualifier was given a parameter value. In this context, no parameter value is permitted.

Action: Remove parameter value and resubmit request.

ARC-01800E No archive event defined for the selection criteria.

Cause: No archive event could be found in the archive catalog that matches the qualifiers specified in the RETRIEVE command.

Action: Use the LIST/DBSPACESET command to verify that the specified dbspace set was archived under the request ID or other qualifiers specified.

ARC-01801E The dbspace set can't be deleted; it is still being referenced.

Cause: The dbspace set specified in a DELETE command still is referenced in at least one request.

Action: Cancel and remove all requests that reference the dbspace set if you really do want to delete the dbspace set.

ARC-01802E A dbspace specified in the list does not exist.

Cause: A dbspace name specified in a DEFINE / DBSPACESET command is not a dbspace known to the database server.

Action: Check the spelling of the dbspace names and use **onstat -d** to list all the dbspaces known to the database server. Either use **onspaces** to define the dbspaces or change the list in the DEFINE command.

ARC-01803E A logfile backup type of either CONTINUOUS or AUTOMATIC must be specified.

Cause: A BACKUP command was specified without either an AUTOMATIC or CONTINUOUS qualifier.

Action: Resubmit the request, including either an AUTOMATIC or CONTINUOUS qualifier.

ARC-01804E Call Informix Technical Support.

Cause: An internal error was detected by ON-Archive.

Action: Refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01805E Error attempting to find INFORMIX logs to back up.

Cause: An error occurred while asking the database server for full log files during a backup request.

Action: See other messages for more details.

ARC-01806E Invalid MODE parameter assigned to MODE qualifier.

Cause: An unrecognized parameter value was specified for the MODE qualifier.

Action: Check the syntax in the reference material or on-line Help.

ARC-01807E Invalid transition between logging modes for database '%s'.

Cause: A MODIFY / DBLOGGING command specified a mode change for the specified database that is illegal.

Action: Check the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for legal transitions and use.

ARC-01808E Error changing the database logging mode.

Cause: An error occurred while ON-Archive was requesting a database logging change.

Action: See other error messages for details.

ARC-01809E Invalid DBSPACESET name.

Cause: A name was specified for a dbspace set that has not been defined in the archive catalog.

Action: Check the spelling of the dbspace-set name or define the dbspace set using the DEFINE / DBSPACESET command.

ARC-01810E Invalid LEVEL value; must be 0, 1, or 2.

Cause: The LEVEL qualifier had a value specified for it that was not allowed.

Action: Resubmit the command with LEVEL set to 0, 1, or 2.

ARC-01811E Error attempting to access DBSPACESET definition.

Cause: An error was encountered when **onarchive** tried to access the definition for the specified dbspace set.

Action: See other error messages, including any in the **/tmp/oncatlgr.out** file.

ARC-01812E Error attempting to get last archive event for prior LEVEL.

Cause: An error was encountered when **onarchive** tried to find all the archive events for the specified dbspace set during a RETRIEVE operation.

Action: See other error messages, including any in the **/tmp/oncatlgr.out** file.

ARC-01813E Error attempting to add new archive event.

Cause: An error was encountered when ON-Archive attempted to insert a new archive event into the archive catalog.

Action: See other messages, including any in the **/tmp/oncatlgr.out** file.

ARC-01814E No logfile unique ID identified.

Cause: The qualifier LOGFILE was specified without a parameter value in a command that requires a unique ID value.

Action: Supply the unique ID value or *.

ARC-01815E Not all dbspaces in specified list are defined for dbspace set.

Cause: One or more dbspace names were specified that were not defined for the dbspace set that was specified in the command.

Action: Check that the dbspaces are in the dbspace set using the LIST / DBSPACESET command.

ARC-01816E The specified dbspace set doesn't exist.

Cause: A command was specified with a dbspace set name that is not defined in the archive catalog.

Action: Verify dbspace-set names with the LIST / DBSPACESET command. Reenter the command with a valid dbspace-set name.

ARC-01817W No match found.

Cause: A retrieve command was specified for a dbspace that is not contained in the archive save set being retrieved.

Action: Use the LIST command of **ondatartr** to list the contents of the save set. The most common occurrence of this error is when the user is trying to restore a dbspace that was created after the archive was done. Use a more current archive or do a whole-system restore.

ARC-01818E ONDATARTR is for catastrophic restore only; the server can't be running.

Cause: A retrieve command was executed from **ondatartr** when the database server was not off-line.

Action: Take OnLine off-line and reexecute the retrieve command.

ARC-01819E List not allowed for logfile.

Cause: The LOGFILE qualifier was specified with a list of parameter values.

Action: Only use a single log unique ID with the LOGFILE qualifier.

ARC-01820E List not allowed for dbspaceset.

Cause: The DBSPACESET qualifier was specified with a list of parameter values.

Action: Reenter the command with only a single dbspace-set name.

ARC-01821E Error connecting to INFORMIX-OnLine server.

Cause: Either **onarchive** or **onautovop** encountered a problem trying to connect to the database server to execute a request.

Action: Verify that OnLine is on-line. If so, verify that other applications and utilities can connect to the database server. If so, refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01822E Error getting root dbspace name from **sysmaster**.

Cause: Either **onarchive** or **onautovop** encountered an error trying to query the name of the root dbspace from the **sysmaster** database.

Action: Verify that OnLine is on-line and that the **sysmaster** database exists. If so, verify that other applications and utilities can connect to the database server. If so, refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01823E No remote shell command for remote tape; check **DBREMOTECMD**, **PATH**.

Cause: When trying to access a remote tape device, ON-Archive would not locate a valid remote shell command in the directory path specified in either the environment variable **DBREMOTECMD** or the environment variable **PATH**.

Action: Set **DBREMOTECMD** to the path where your remote UNIX shell command can be found and then resubmit the command.

ARC-01824E Cannot open pipe for remote device I/O.

Cause: An attempt to use a remote tape device resulted in a UNIX pipe error.

Action: See the accompanying UNIX error numbers for more information. Verify that the **dd** UNIX command is accessible to the user executing ON-Archive.

ARC-01825E Cannot fork shell process for remote device I/O.

Cause: An attempt to use a remote tape device resulted in a UNIX fork error.

Action: See the accompanying UNIX error numbers for more information. Verify that the **dd** UNIX command is accessible to the user executing ON-Archive.

ARC-01826E Incorrect server version or restore level.

Cause: During the execution of a retrieve operation, **onarchive** or **ondatartr** got a version or level mismatch between what was expected by the software and what was listed on the tape.

Action: Use the LIST command of **ondatartr** to list the version and level number on the volume. Reexecute the retrieve command with the correct volume.

ARC-01827E Cannot define a remote node for a DISK device.

Cause: The user attempted to define a vset with a driver of DISK and a DEVICE_TYPE that includes a physical device with a node specification (for example, **remsite:/dev/rst0**). This error also occurs if the user attempts to use a remote device for DISK when using **ondatartr**.

Action: Either remove the node specification from the physical-device specification or use a different device.

ARC-01828E First page of archive isn't archive header.

Cause: During processing of RETRIEVE/DBSPACESET command, **ondatartr** could not find the archive header.

Action: The operator might have mounted the wrong volume. Use the LIST command of **ondatartr** to verify the contents of the tape. The volume might be corrupted. Try using another copy of the volume.

ARC-01829E The star is not permitted for USER in DEFINE or MODIFY.

Cause: When the USER qualifier is used in a DEFINE or MODIFY command, the * syntax is not allowed.

Action: Specify a user or user list and resubmit the command.

ARC-01830E Archive/Backup to remote device must use APART.

Cause: An archive or backup request was executed to a remote tape device, but the APART qualifier was not specified.

Action: Either add the APART qualifier and resubmit the command or use a different device and resubmit the command.

ARC-01831E Must be in restore mode to do logical restore.

Cause: The logical restore (RETRIEVE / LOGFILE) was attempted when the database server was not in restore mode.

Action: Always precede RETRIEVE / LOGFILE by RETRIEVE / DBSPACESET.

ARC-01832E The star is not permitted for DBSPACESET or DBSPACE in DEFINE.

Cause: A DEFINE / DBSPACESET command was specified where either the dbspace-set name or the dbspace-name list was *.

Action: Provide a specific name for the dbspace set and a list of specific dbspace names for the dbspace list. You can also perform a full-system archive using * with the ARCHIVE command.

ARC-01833E This vset already exists in the catalog.

Cause: A DEFINE / VSET command was specified with the vset name of a vset that already is defined.

Action: Check the name of the vset you are trying to define. Do a LIST / VSET command to list all the vsets that are already defined.

ARC-01834E Remote shell error messages: %s

Cause: A remote shell was executed to do I / O to a remote tape device, but operating-system errors occurred.

Action: The operating-system errors should provide information about what caused the error during the remote tape I / O.

ARC-01835E Max space is too small, minimum is 16.

Cause: A volume was defined with a MAX_SPACE smaller than 16, which is the smallest buffer of data that is written by ON-Archive.

Action: Specify a larger value of MAX_SPACE.

ARC-01836E Invalid delta date format %s

Cause: A qualifier was specified that takes a delta date value, and that value does not match the allowable format for delta dates.

Action: Correct the format of the delta date being specified.

ARC-01837E MAX_SPACE not allowed for a TAPE vset.

Cause: A volume was specified for a TAPE vset with a nonzero MAX_SPACE value.

Action: Do not specify a MAX_SPACE value for volumes of TAPE vsets.

ARC-01838E Invalid serial number. Please consult the Installation Instructions.

ARC-01839E Dbspaceset name of save set doesn't match dbspaceset being retrieved.

Cause: The **ondatartr** program has detected a discrepancy between the dbspace set it is being asked to retrieve and the dbspace set that was archived in the specified save set. Probably, the wrong save-set number was specified.

Action: Use the **ondatartr** LIST command to list the contents of the save sets. Then perform the retrieve request again with the correct dbspace-set and save-set combination.

ARC-01840E Attribute value not found in the return from the query.

Cause: The **sysmaster** database might be inconsistent.

Action: Running **arc_purge.sql** might fix the problem. If it does not fix the problem, recatalog the **sysmaster** database. If the problem persists beyond these steps, refer to Appendix B, "Trapping Errors," in the *INFORMIX-OnLine Dynamic Server Administrator's Guide* for more diagnostics and then contact Informix Technical Support with this information.

ARC-01841E Ambiguous command, %s requires more qualifiers.

Cause: Not enough required qualifiers were recognized from the command line to identify which command was given to ON-Archive.

Action: Review the syntax for the command in the documentation and reissue the command with appropriate modifications.

ARC-01842E Invalid parameter format.

Cause: Invalid parameter format was used in an ON-Archive command.

Action: Review the syntax for the command in the documentation and reissue the command with the correct parameter format.

ARC-01843E Must be user **informix** or **root** to run this program.

Cause: Only user **informix** or **root** are allowed to run this program. A user other than user **informix** or **root** is attempting to run **oncatlgr**, **onkeymgr**, or **ondatartr**.

Action: Run the program as user **informix** or **root**.

ARC-01844W WARNING: Directory is invalid or does not exist.

Cause: While creating a disk volume, **onarchive** noted that the last node in the VIRTUAL path did not already exist.

Action: **onarchive** continues and creates the volume and the missing node, so no action is required, but you might wish to verify that you specified the VIRTUAL path as you intended.

ARC-01845W Initializing this media may overwrite an ON-Archive volume.

Cause: During the creation of a tape volume, it was determined that the tape being used for a new volume was previously used as an ON-Archive volume. Any data on the volume becomes inaccessible once the volume is reinitialized.

Action: This warning verifies that reinitializing this tape is really what you want to do.

ARC-01846E Error received from Informix server while processing archive or backup.

Cause: This message indicates that the error message following this one is from the database server, rather than from the ON-Archive product.

Action: The message that follows this one should indicate why the database server is rejecting the attempt to do this action. Correct the situation and try again.

ARC-01847W Possible inconsistency between catalog and contents of disk vset %s.

Cause: For the vset listed, there is a discrepancy between what the catalog has listed for save sets on the vset and what is actually on the vset. An additional file in the **/tmp** directory gives the full status of the save set.

Action: Check the status file in the **/tmp** directory for specifics on which save sets are missing or present. If a save set is on the disk, but not in the catalog, perhaps the save set needs to be recataloged (if you wish to use it) or deleted (if you will not be using it). If save sets are listed in the catalog that are not on disk, you must remove the requests for those save sets from the catalog or you must restore the files if they were saved elsewhere.

ARC-01848E **INFORMIXDIR** not set in environment

Cause: The **INFORMIXDIR** environment variable, which specifies the path to the subdirectories where the OnLine product files reside, is not set.

Action: Set the **INFORMIXDIR** environment variable to the directory path where you installed OnLine.

ARC-01849E A dbspace in the list is a temporary dbspace and can't be archived.

Cause: One of the dbspaces listed in the command issued is defined as a temporary dbspace, and ON-Archive will not archive it.

Action: Reissue the command without the temporary dbspace in the list.

ARC-01850E Cannot modify the vset class from SYSTEM to USER if volumes are defined.

Cause: ON-Archive does not allow the class of a vset to be changed from SYSTEM to USER if volumes have been defined for that vset. This prevents ON-Archive from later denying access to users who wrote data to the vset before its class was changed.

Action: If it is important that the class of the vset be changed, copy any existing data to another vset, delete all the volumes in that vset, then copy the data back.

ARC-01851E Cannot specify /TRANSIT for an archive or backup to a disk vset.

Cause: The user entered an ARCHIVE or BACKUP command that specified writing to a vset on disk and also specified the /TRANSIT qualifier. This action is not supported by ON-Archive.

Action: Create enough volumes in the desired archive/backup destination disk vset that are large enough to hold the data, or change the archive/backup command to use a tape vset as the destination.

The ontape Utility

Section VI

Chapter

12

Configuring ontape

This chapter explains how to set the INFORMIX-Universal Server configuration parameters that the **ontape** utility uses to create archives of Universal Server data and back up logical-log files. The **ontape** utility provides an alternative Universal Server recovery system to ON-Archive. See "What Are the Differences Between ON-Archive and ontape?" on page 1-9 for a description of how **ontape** differs from ON-Archive.

This chapter describes the following tasks:

- Setting **ontape** configuration parameters
- Checking **ontape** configuration parameters
- Changing **ontape** configuration parameters

The following chapter, Chapter 13, "Using ontape," describes how to use the **ontape** utility to create an archive of your Universal Server data and back up your logical-log files.

Warning: *The **ontape** utility and ON-Archive produce incompatible archive tapes! You cannot create an archive with **ontape** and restore it with ON-Archive. Also, do not try to restore some logical logs created with one tool and then restore other logical logs from a tape created with the other tool.*

Setting the ontape Configuration Parameters

The **ontape** utility uses six parameters in the Universal Server ONCONFIG file to create archives and back up logical-log files. The ONCONFIG file is located in the **$INFORMIXDIR/etc** directory. You specify that file in the **ONCONFIG** environment variable. See the *Informix Guide to SQL: Reference* for a description of the **ONCONFIG** environment variable and instructions on how to set it.

The six ONCONFIG parameters that **ontape** uses divide into two sets. The first set specifies the characteristics of the tape device and tapes for archives; the second set specifies the characteristics of the tape device and tapes for logical-log file backups.

The following list shows archive tape devices and their associated tape parameters:

TAPEDEV is the tape device used for archiving.

TAPEBLK is the block size of the tapes used for archiving, in kilobytes.

TAPESIZE is the size of the tapes used for archiving, in kilobytes.

The following list shows the logical-log tape devices and their associated tape parameters:

LTAPEDEV is the logical-log tape device.

LTAPEBLK is the block size of tapes used for logical-log backups, in kilobytes.

LTAPESIZE is the size of tapes used for logical-log backups, in kilobytes.

The following sections contain information about how to set the tape-device, tape-block-size and tape-size parameters for both archives and logical-log backups.

Setting the Tape-Device Parameters

You must consider the following points when you assign values to TAPEDEV and LTAPEDEV:

- Use separate devices, when possible.
- Use symbolic links.
- Specify remote devices.
- Specify **/dev/null**.
- Rewind tape devices.

The following sections explain each of these points.

Specify Separate Devices for Logical-Log Backups and Archiving

When possible, the LTAPEDEV and TAPEDEV parameters in the ONCONFIG file must each specify a different device. When specifying separate devices for archives and logical-log backups you can schedule archives and logical-log backups independently of each other. You can create an archive on one device at the same time you continuously back up the logical-log files on the other.

When the LTAPEDEV and TAPEDEV parameters specify the same device, the logical log can fill and cause Universal Server to stop processing during an archive. When this happens, you face limited options. You can either abort the archive to free the tape device and back up the logical-log files or leave normal processing suspended until the archive completes.

Precautions to Take When You Use One Tape Device

When only one tape device exists and you want to create archives while Universal Server is in on-line mode, you can take the following precautions:

- Configure Universal Server with a large amount of logical-log space through a combination of many log files, or large log files. (See the *INFORMIX-Universal Server Administrator's Guide*.)
- Store all explicitly created temporary tables in a dedicated dbspace and then drop the dbspace before archiving.
- Create the archive when low database activity occurs.
- Free as many logical-log files as possible before you begin the archive.

The logical log can fill up before the archive completes. The archive synchronizes with an Universal Server checkpoint. You could require an archive to wait for a checkpoint to synchronize activity, but the checkpoint cannot occur until all virtual processors exit critical sections. When Universal Server processing suspends because of a full logical-log file, the virtual processors cannot exit their critical sections and a deadlock results.

Consider Using Symbolic Links to Specify Tape Devices

You can specify the values of LTAPEDEV and TAPEDEV as symbolic links. Using symbolic links enables you to switch to other tape or tape-compatible devices without changing the pathname in the ONCONFIG file. For example, you can specify the following symbolic link for tape device **/dev/rst0**:

```
ln -s /dev/rst0 /dbfiles/logtape
```

Then, when you set the LTAPEDEV configuration parameter as shown in the following example:

```
LTAPEDEV /dbfiles/logtape
```

you can switch to a different device without changing the LTAPEDEV parameter. You only need to change the symbolic link, as shown in the following example:

```
ln -s /usr/backups /dbfiles/logtape
```

A user with one tape device could want to redirect a logical-log backup to a disk file while using the tape device for an archive.

Syntax to Specify a Remote Device

You can specify an archive or logical-log backup device attached to another host computer, and you can perform an archive or logical-log backup across your network to a remote device—that is, a device attached to another host computer. You must not do a continuous backup to a remote device. To specify a tape device on another host computer, use the following syntax:

```
host_machine_name:tape_device_pathname
```

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

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

See "Tape Size for Remote Devices" on page 12-8 for information on the tape size for remote devices.

Using /dev/null for a Tape Device

Informix recommends that you do not use **/dev/null** as the device when archiving. However, when you specify **/dev/null** as an archive tape device, you can avoid the overhead of a level-0 archive that is required after some operations—like changing the logging status of a database, for example. Obviously, you cannot restore Universal Server data from an archive to **/dev/null**.

As described in "Do You Need to Back Up the Logical-Log Files?" on page 13-30, you can specify **/dev/null** as a tape device for logical-log backups when you decide that you do not need to recover transactions from the logical log.

When you specify the tape device as **/dev/null**, block size and tape size are ignored.

Warning: *When you set the ONCONFIG parameter LTAPEDEV to* ***/dev/null****, Universal Server marks the logical-log files as backed up as soon as they become full, effectively discarding logical log information. This is true for ON-Archive as well as* ***ontape****, even though ON-Archive does not use LTAPEDEV to specify devices. See "When You Do Not Need to Recover" on page 6-5 for an explanation of the impact of setting LTAPEDEV to* ***/dev/null*** *for ON-Archive.*

Tape Devices Must Rewind Before Opening and on Closing

ON-Archive requires that tape devices must rewind before opening and on closing. Before reading from or writing to a tape, Universal Server performs a series of checks that require the rewind.

Specifying the Tape-Block-Size Parameters

Specify the block-size parameters as the largest block size, in kilobytes, that your tape device permits.

When you set the tape parameter to **/dev/null**, the corresponding block size is ignored.

Universal Server does not check the tape device when you specify the block size. Verify that the tape device can read the block size that you specified. If not, you cannot restore the tape.

Specifying the Tape-Size Parameters

The number of blocks specify tape sizes. They specify the maximum amount of data that you can write to a tape.

When you specify the tape device as **/dev/null**, the corresponding tape size is ignored.

Tape Size for Remote Devices

When you perform a continuous logical-log backup, the amount of data written to the tape is the smaller of LTAPESIZE and the following formula:

```
(sum of space occupied by all logical log files on disk) -
(largest logical-log file)
```

This ensures that the I/O to the remote device completes and you free the logical-log files before a log-full condition occurs.

Checking ontape Configuration Parameters

To examine your ONCONFIG file (the file specified in **$INFORMIXDIR/etc/$ONCONFIG**), execute **onstat -c** while Universal Server is running.

Do not use ON-Monitor to look at the ONCONFIG file. The configuration displayed using ON-Monitor (Status menu, Configuration option) is a copy of the *current* Universal Server configuration, and that can differ from the values stored in the configuration file when you made changes after you initialized Universal Server.

Changing ontape Configuration Parameters

This section provides general information on changing **ontape** configuration parameters such as who can change them, when you can make the changes, whether they require a level-0 archive, and so on. This section also describes how to change **ontape** parameters using either ON-Monitor or a text editor.

About Changing ontape Parameters

Bear in mind the following points when you change any of the **ontape** configuration parameters:

- Who can make the changes?
- When can you make changes?
- Create a level-0 archive after you make changes.
- Verify that the tape device can read the tape block size specified.

The following sections explain each of these points.

Who Can Change ontape Parameters?

When you log in as either user **informix** or **root**, you can change the value of **ontape** configuration parameters from within ON-Monitor or when using an editor.

When Can You Change ontape Parameters?

You can change the values of **ontape** parameters while Universal Server is in on-line mode. The change takes effect immediately. As explained in the following sections, however, you must make additional considerations when you change either the TAPEDEV parameter or the LTAPEDEV parameter to **/dev/null**.

Changing TAPEDEV to /dev/null

The **ontape** utility reads the value of the TAPEDEV parameter at the start of processing. When you set TAPEDEV to **/dev/null** and request an archive, Universal Server bypasses the archive but still updates the dbspaces with the new archive timestamps. When you set TAPEDEV to **/dev/null,** you must do it before you start **ontape** to request the archive. No problems exist when you change TAPEDEV to **/dev/null** while Universal Server is in on-line mode and **ontape** is not running.

Changing LTAPEDEV to /dev/null

Take Universal Server off-line before you change the value of LTAPEDEV to **/dev/null**. When you make the change while Universal Server operates in either quiescent or on-line mode, you can create a situation where you back up one or more log files but do not free them. This situation can interrupt processing because Universal Server stops when it finds that the next logical-log file (in sequence) is not free.

When you set LTAPEDEV to **/dev/null**, the Universal Server database server frees the logical logs without requiring that you back up those logs. The logical logs do not get marked as free, but the Universal Server database server can reuse them.

Create a Level-0 Archive After You Change Tape Device Parameters

To ensure a proper restore, you must create a level-0 archive immediately after you change any of the archive or logical-log file backup tape device parameters, unless you change the value to **/dev/null**. You create the level-0 archive for two reasons.

- The Universal Server restore procedure with **ontape** cannot switch tape devices as it attempts to read the logical-log backup tapes. When the physical characteristics of the log file tapes change during the restore, either because of a new block size or tape size, the restore fails.
- The restore fails when the tape device specified as TAPEDEV or LTAPEDEV at the time of the level-0 archive is unavailable when the restore begins.

Verify That the Tape Device Can Read the Block Size Specified

Universal Server does not check the tape device when you specify the block size. Verify that the tape device specified in TAPEDEV and LTAPEDEV can read the block size you specify for their block-size parameters. If not, you cannot restore the tape.

Changing ontape Parameters Using ON-Monitor

You can use ON-Monitor to change the **ontape** parameters for either archiving or logical-log backup.

Changing Archive Tape Parameters Using ON-Monitor

To start ON-Monitor, use the following command:

```
% onmonitor
```

Select the Archive menu, Tape-Parameters option to change the values of TAPEDEV, TAPEBLK, and TAPESIZE. ON-Monitor displays the current values.

Enter the new full pathname value for the logical log tape device in the **Log Tape Device** field. Enter new values in the device **Block Size** and **Tape Size** fields, when appropriate.

The change takes effect immediately.

Perform a level-0 archive, as explained in "Create a Level-0 Archive After You Change Tape Device Parameters" on page 12-10.

Changing Logical-Log Backup Tape Parameters Using ON-Monitor

Select the Logical-Logs menu, Tape-Parameters option to change the values of LTAPEDEV, LTAPEBLK, and LTAPESIZE. ON-Monitor displays the current values.

Enter the new full pathname value for the logical log tape device in the `Log Tape Device` field. Enter new values in the device `Block Size` and `Tape Size` fields, when appropriate.

The change takes effect immediately.

Perform a level-0 archive, as explained in "Create a Level-0 Archive After You Change Tape Device Parameters" on page 12-10.

Changing Archive-Device Parameters Using an Editor

You can use an editor to change the **ontape** parameters for either archiving or logical-log backup using an editor.

Changing Archive-Tape Parameters Using An Editor

To change the value of TAPEDEV, TAPEBLK, and TAPESIZE from the command line, use an editor to edit your ONCONFIG file. Change the value of TAPEDEV (and TAPEBLK and TAPESIZE, when appropriate). Save the file.

The change takes effect immediately.

Perform a level-0 archive, as explained in "Create a Level-0 Archive After You Change Tape Device Parameters" on page 12-10.

Changing Logical-Log Backup Tape Parameters Using an Editor

To change the value of LTAPEDEV, LTAPEBLK, and LTAPESIZE from the command line, use an editor to edit your ONCONFIG file. Change the value of LTAPEDEV (and LTAPEBLK and LTAPESIZE, when appropriate). Save the file.

The change takes effect immediately.

Perform a level-0 archive, as explained in "Create a Level-0 Archive After You Change Tape Device Parameters" on page 12-10.

Chapter

13

Using ontape

This chapter describes how to use **ontape** as your recovery system for the INFORMIX-Universal Server. It describes how to use **ontape** to perform the following tasks:

- Create archives
- Back up logical-log files
- Restore data

Chapter 1, "What Is a Universal Server Recovery System?," explains the role of each of these tasks in the Universal Server recovery system.

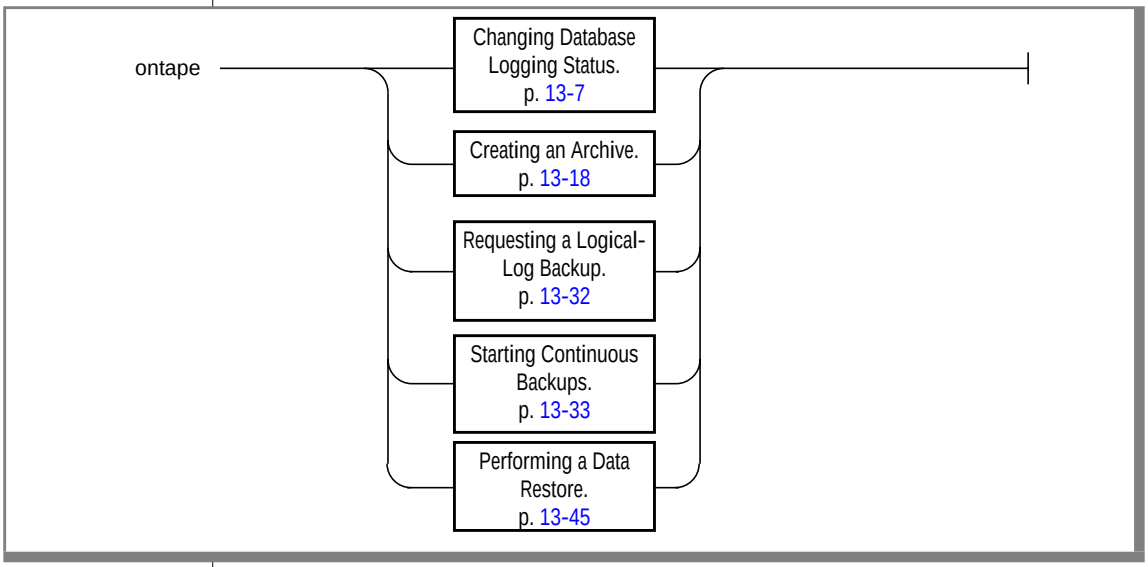
You can also use **ontape** to change the database logging status of Universal Server databases. The *INFORMIX-Universal Server Administrator's Guide* describes this task.

Warning: *ON-Archive and* ***ontape*** *create incompatible tapes. You must use either ON-Archive or* ***ontape****, not both. See "What Are the Differences Between ON-Archive and ontape?" on page 1-9 for a comparison of ON-Archive and* ***ontape****.*

Warning: *The* ***ontape*** *utility does not include default values for user interaction, nor does it support retries. When it expects a yes/no response, then it assumes that any response not recognized as a "yes" is "no".*

Syntax of ontape

The **ontape** utility provides options that enable you to change the logging status of a database, create an archive, back up the logical-log files, back up the logical-log files on a continuous basis, and restore Universal Server data from an archive and logical-log backup. The following syntax diagram illustrates the basic syntax of the **ontape** utility.



When you need more than one tape during an archive or logical-log backup, **ontape** prompts for each additional tape.

Warning: *Do not start* ***ontape*** *in background mode (that is, using the UNIX* ***&*** *operator on the command line). The* ***ontape*** *utility is interactive and prompts you to mount new tapes when necessary. You could also need to provide input from the terminal or window. When you execute* ***ontape*** *in background mode, you can miss prompts and delay an operation.*

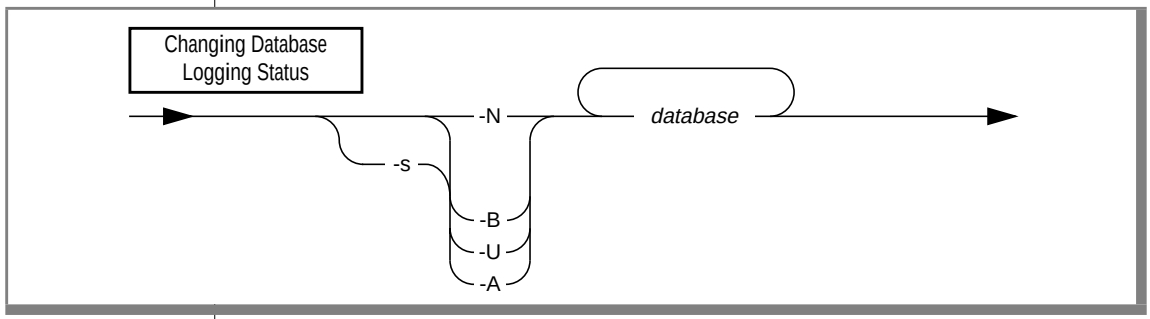
Exit Codes

The **ontape** utility has the following two exit codes:

0 indicates a normal exit from **ontape**.

1 indicates an exception condition.

Changing Database Logging Status



When you add logging to a database, you must create a level-0 archive before the change takes effect.

-A directs **ontape** to change the status of the specified database to ANSI-compliant logging.

-B directs **ontape** to change the status of the specified database to buffered logging.

database is the name of the database. The database name cannot include a database server name.

-N directs **ontape** to end logging for the specified database.

-s initiates an archive.

-U directs **ontape** to change the status of the specified database to unbuffered logging.

These **ontape** command options are similar to those available from the ON-Monitor Logical-logs menu, Database option.

See the *INFORMIX-Universal Server Administrator's Guide* and "Scheduling Archives" on page 13-10 for considerations about changing the logging status of a database.

Creating an Archive

This section explains how to plan for and create archives of your Universal Server data. It discusses the following topics:

- Considerations before you create an archive
- Archive levels
- Performing an archive
- Details of an archive

What Are Archive Levels?

When Universal Server manages a large tape of data, it does not always make sense to archive all the data each time you create an archive. For example, when some of your information changes quite a bit, but some remains very stable, it seems inefficient to archive the stable information every time you archive the volatile information.

To provide a more flexible archive environment, Universal Server supports three archive *levels*:

Level-0 archives all used pages.

Level-1 archives all changes after the last level-0 archive.

Level-2 archives all changes after the last level-1 archive.

The following sections explain the archive levels.

You need to plan your archive schedule carefully to avoid long delays for archiving or restoring data.

Level-0 Archives

A level-0 archive is the baseline archive. It contains a copy of every used disk page (dbspace and blobspace) that you need to restore the Universal Server database server to its state at that time. When a fire of flood, for example, completely destroys a computer—you need a level-0 archive to completely restore Universal Server data on the replacement computer.

For on-line archives, the data on the archive tape reflects the contents of the dbspaces and blobspaces at the time the level-0 archive began. (The time the archive started could reflect the last checkpoint before the archive started, when no database activity occurred between the checkpoint and the archive.)

A level-0 archive can consume lots of time because Universal Server must write all the pages to tape.

Level-1 Archives

A level-1 archive contains a copy of every changed page containing data and system overhead information after the last level-0 archive. All data copied to the archive reflects the state of the data at the time the level-1 archive began. A level-1 archive usually takes less time than a level-0 archive because you copy only part of the data, that Universal Server manages, to the archive tape.

Level-2 Archives

A level-2 archive contains a copy of every changed page containing data and system-overhead information after the last level-1 archive. All data copied to the archive reflects the state of the data at the time the level-2 archive began.

A level-2 archive after a level-1 archive usually takes less time than another level-1 archive because only the changes made after the last level-1 archive (instead of the last level-0) get copied to the archive tape.

Scheduling Archives

You must make a regular schedule for creating archives. Level-1 and level-2 archives are optional in your schedule, but level-0 archives are not. At the very least, the following administrative changes require a level-0 archive as part of the procedure. Consider waiting to make these changes until your next regularly scheduled level-0 archive.

- Changing TAPEDEV or LTAPEDEV from **/dev/null** requires an archive after the you make the change.
- Adding logging to a database requires an archive after you add logging.
- Adding a dbspace or blobspace requires that you can archive the space before you can restore it with anything less than a full-system restore.
- Starting mirroring for a dbspace that contains logical-log files requires an archive after the change to initiate mirroring.
- Adding a logical-log file requires an archive afterward to make the log file available.
- Dropping a logical-log file requires an archive after you drop the log file.
- Moving one or more logical-log files requires an archive after you drop or add the logical-log file.
- Changing the size or location of the physical log requires an archive after you reinitialize shared memory.
- Dropping a chunk requires an archive before you can reuse the dbspace containing that chunk.

Figure 13-1 shows three very different archive schedules, ranging from one that creates archives very frequently to one that does not.

Figure 13-1
Examples of Archive Schedules

Level	Daily Schedule	Weekly Schedule	Monthly Schedule
Level-0	Every night	Sundays	Once a month
Level-1	At lunch	Wednesdays	Once a week
Level-2	Every two hours	Mondays, Tuesdays, Thursdays, Fridays	Once a day

You can adopt a schedule like one of the preceding schedules or develop a schedule of your own. The schedule you develop depends on how much time you want to devote to making archives, how much time you can devote to a restore, the number of available tape drives, and the availability of an operator for making archives.

Determining Your Priorities for the Schedule

Each of the following considerations affect the archive schedule you create for your environment:

- Do you need to minimize the time for a restore?
- Do you need to minimize the time to create an archive?
- Do you need to create archives while Universal Server operates in on-line mode?
- Do you need to use the same tape drive to create archives and back up logical-log files?
- Is the operator periodically unavailable?

Estimating the Time Required for an Archive

You must consider several variables when you estimate the time it takes to perform an archive. Each of the following items has an impact on the time needed to complete an archive:

- Overall speed of the tape device, including operating-system overhead
- Level of the archive
- Size of the archive
- Amount and type of database activity during the archive
- Amount and type of database activity in the period after the last archive
- Alertness of the operator to tape-changing demands
- Whether you archive different dbspaces concurrently, using multiple tape drives.

The best approach to estimating the time needed to complete an archive is to create an archive and try to gauge the time for subsequent archives using the first archive as a basis for comparison.

Minimizing Archive Size

The size of a level-1 archive is a function of the time and amount of update activity after your level-0 archive. The more often you create level-0 archives, and the less updating between archives, the smaller each level-1 archive becomes. Level-2 two archives can also be smaller when level-1 archives are more frequent and require less updating between them.

Minimizing the Time for an Archive

To reduce the duration of an archive, you can also reduce the number of data pages that you must archive. You can reduce the number of data pages that you must archive when you manage space for temporary tables in one of the following ways:

- Create a temporary dbspace, as described in the *INFORMIX-Universal Server Administrator's Guide*, and store your temporary tables there. Universal Server ignores any tables stored in a temporary dbspace during an archive.
- Create a normal dbspace where you can store all temporary tables; then, drop the dbspace before you create an archive
- Drop all temporary tables before you archive the dbspaces in which they reside.

You only need to do a level-0 archive after creating a dbspace when you plan to restore it. When you do not create a level-0 archive for the dbspace and a critical media failure occurs, one of the following two things happens:

- The dbspace gets marked as disabled and the chunks go down
- The dbspace gets recovered when you replay the logs because you logged the statement that created the dbspace.

When you leave the dbspace in a disabled state, you can drop the dbspace and re-create it.

How to Minimize the Time for a Restore

The following list shows factors that affect the time required to perform a restore:

- Size and number of archives.

  The minimum number of archives needed to restore is one level-0 archive. The maximum number is three, one of each archive level.
- Amount of data you intend to restore.

- Size and number of logical-log files after the last archive.

  More log files take longer to restore.

- Type of restore.

  When you perform a full-system restore, you can restore some dbspaces first, while Universal Server is off-line. Then, when Universal Server comes on-line, those dbspaces remain available while other dbspaces are being restored. This type of restore increases the availability of some dbspaces but also increases the total restore time. When your logical log becomes full, Universal Server suspends processing until you back it up. This means either that you must abort the archive or processing must remain suspended until you complete the archive and you back up the logical-log files.

You can consider the following strategy to minimize the time needed to restore an Universal Server database server:

- Create a level-0 archive as often as you can, perhaps every three days.
- Create a level-1 archive daily.
- Do not use level-2 archives.

The time required for any possible restore is limited to the time needed to read and process the following data:

- A level-0 archive of the dbspace(s) being restored
- A level-1 archive, representing from one to three days' activity in the dbspaces being restored
- Logical-log files, representing less than a day's work in the dbspace or dbspaces being restored

Before You Create an Archive

Take the following precautions when you create an archive:

- Avoid using temp tables during heavy activity.
- Make sure you make sufficient logical-log space to create an archive.
- Keep a copy of your ONCONFIG file.
- Verify data consistency.
- Run Universal Server in the appropriate mode.
- Plan for operator availability.
- Synchronize with other administrative tasks.
- Do not use background mode.
- Label tapes appropriately.

The following sections address each of these topics.

Avoid Using Temp Tables During Heavy Activity

When you create a temp table during an archive while using the **ontape** utility, that table is placed in DBSPACETEMP. When heavy activity occurs during the archive process, the temp table can keep growing and can eventually fill up DBSPACETEMP. When this situation occurs, the archive aborts and your monitor displays a NO FREE DISK error message.

Make Sure Enough Logical Log Space Exists

When the total available space in the logical log (all the logical-log files) amounts to less than half a single log file, Universal Server does not create an archive. You must back up the logical-log files and attempt the archive again.

You cannot add a logical-log file or mirroring during an archive.

When you use only one available tape device, make sure you back up all your logical-log files before you start your archive to reduce the likelihood of filling the logical log during the archive.

Keep a Copy of Your ONCONFIG File

Keep a copy of the current ONCONFIG file when you create a level-0 archive. You need this information to restore Universal Server data from the archive tape.

Verify Consistency Before a Level-0 Archive

To ensure the integrity of your archives, periodically verify that all Universal Server data and overhead information is consistent before you create a full-system level-0 archive. You need not check this information before every level-0 archive, but Informix recommends that you keep the necessary tapes from the most-recent archive created immediately after Universal Server was verified as consistent. See the *INFORMIX-Universal Server Administrator's Guide* for information on consistency checking.

Archives and Modes

You can create an archive while Universal Server operates in on-line or quiescent mode. The terminal you use to initiate the archive command is dedicated to the archive (displaying messages) until the archive completes. Once you start an archive, Universal Server must remain in the same mode until the archive finishes; changing the mode terminates the archive activity.

What Is an On-line Archive?

An on-line archive is an archive that you create while Universal Server is in on-line mode. You can use this type of archive when you want your Universal Server database server accessible while you create the archive.

Some minor inconveniences can occur during on-line archives. An on-line archive can slow checkpoint activity, and that can contribute to a loss in performance. However, this decline in performance is far less costly than the time that you lose when users were denied access to Universal Server during an archive.

During an on-line archive, allocation of some disk pages in dbspaces and blobspaces can temporarily freeze. Disk-page allocation in dbspaces and blobspaces is blocked for one chunk at a time until you archive the used pages in the chunk.

What Is a Quiescent Archive?

You create a quiescent archive while Universal Server is in quiescent mode. You use quiescent archives when you want to eliminate partial transactions in an archive.

Do not use quiescent archives when users need continuous access to the databases that Universal Server manages.

Ensure That the Operator Is Available

Keep an operator available during an archive to mount tapes as prompted.

An archive could take several reels of tape. When an operator is not available to mount a new tape when one becomes full, the archive waits. During this wait, when the archive is an on-line archive, the physical log space could fill up, and that causes Universal Server to abort the archive. Thus, make sure an operator is available.

Labelling Tapes Created Using ontape

When you label tapes created using **ontape**, the label must include the following information:

- Archive level
- Date and time
- Tape number that **ontape** provides

The following example shows what a label can look like:

```
Level 1: Wed Nov 27, 1993 20:45 Tape # 3 of 5
```

Each archive begins with its first tape reel numbered 1. You number each additional tape reel consecutively thereafter. You number a five-tape archive 1 through 5. (Of course, it is possible that you could not know that it is a five-tape archive until it is finished.)

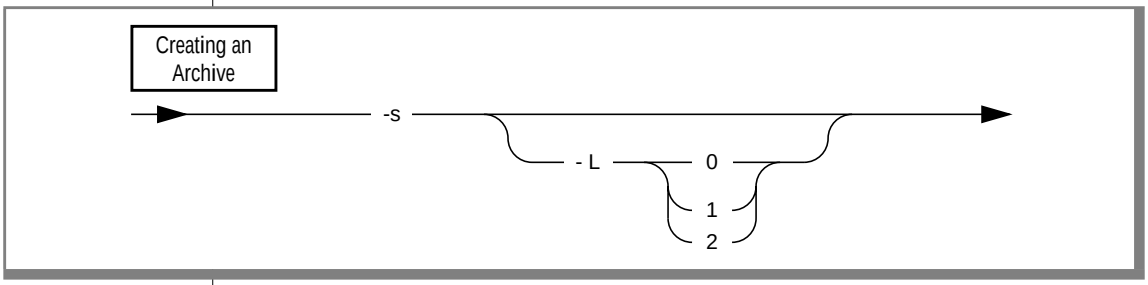
Performing an Archive

You must take the following steps immediately before you begin an archive:

- Place a write-enabled tape on the tape-drive device that TAPEDEV specifies.
- Put the device on-line with the appropriate operating-system command.
- Place Universal Server in on-line or quiescent mode

Do not store more than one archive on the same tape; begin every archive with a different tape. (Often, an archive spans more than one tape.)

To create an archive, use the **-s** option of the **ontape** command.



-s directs **ontape** to create an archive.

-L directs ontape to create an archive of the level specified.

The **-s** option prompts you to supply the archive level—0, 1, or 2—that you wish to create. When you use the **-L** option to specify the archive level as part of the command, you can avoid being prompted for it.

An archive can require multiple tapes. After a tape fills, **ontape** rewinds the tape, displays the tape number for labelling, and prompts the operator to mount the next tape, when you need another one. Follow the prompts for labelling and mounting new tapes. A message informs you when you complete the archive.

Examples

Execute the following command to start an archive without specifying a level:

```
% ontape -s
```

You can use the **-L** option to specify the level of the archive as part of the command, as illustrated in the following example:

```
% ontape -s -L 0
```

When you do not specify the archive level on the command line, **ontape** prompts you to enter it. Figure 13-2 illustrates a simple **ontape** archive session.

```
% ontape -s
Please enter the level of archive to be performed (0, 1,
  or 2) 0

Please mount tape 1 on /dev/rst0 and press Return to
  continue ...
16:23:13 Checkpoint Completed: duration was 2 seconds
16:23:13 Level 0 Archive started on rootdbs
16:23:30 Archive on rootdbs Completed.
16:23:31 Checkpoint Completed: duration was 0 seconds

Please label this tape as number 1 in the arc tape sequence.
This tape contains the following logical logs:

 3

Program over.
```

Figure 13-2
Example of a Simple Archive Created with ontape

When the Logical-Log Files Fill During an Archive

When the logical log fills during an archive, the console displays a message and the archive suspends normal processing. How you handle the logical log filling depends on whether you can use one or two tape devices.

When You Can Use Two Tape Devices

When you can use two tape devices with Universal Server, log in as user **informix** at a free terminal.

Verify that LTAPEDEV and TAPEDEV specify different pathnames that correspond to separate tape devices. When they do, back up the logical-log files. See "Creating an Archive" on page 13-8.

When LTAPEDEV and TAPEDEV are identical, assign a different value to the logical-log tape device (LTAPEDEV) and initiate a logical-log-file backup. This option is only a solution, however, when the new value of LTAPEDEV is compatible with the block size and tape size used to create earlier logical-log file backups. (All tapes must reflect the physical characteristics specified at the time of the most-recent level-0 archive.) Otherwise, your options are to either leave normal Universal Server processing suspended until the archive completes or cancel the archive.

When Only One Tape Device Is Available

When you create an archive with the only available tape device, you cannot back up any logical-log files until you complete the archive. When the logical-log files fill during the archive, normal Universal Server processing halts. You can either abort the archive to free the tape device and back up the logical logs to continue processing or leave normal processing suspended until the archive completes.

You can take steps to prevent this situation. The section "Precautions to Take When You Use One Tape Device" on page 12-5 describe these steps.

When an Archive Terminates Prematurely

When you cancel or interrupt an archive, sometimes the archive progresses to the point where you can consider it complete. When listed in the monitoring information, as described "Monitoring Archive History" on page 13-21, you know the archive completed.

Monitoring Archive History

You can monitor the history of your last full-system archive using ON-Monitor or **oncheck.**

Monitoring Archive History Using ON-Monitor

After starting ON-Monitor, select the Status menu, Archive option.

The display lists the last level-0, level-1, and level-2 archives. After you create a new level-1 archive, earlier level-2 archives no longer appear.

The following information displays for each archive:

- Archive level (0, 1, or 2)
- Date and time of the archive
- Id number of the logical-log file that was current when the archive began

Monitoring Archive History Using oncheck

Execute **oncheck -pr** to display reserved-page information for the root dbspace. The last pair of reserved pages contains the following information for the most recent archive:

- Archive level (0, 1, or 2)
- Effective date and time of the archive
- Timestamp describing when the archive began (expressed as a decimal)
- Id number of the logical log that was current when the archive began
- Physical location in the logical log of the checkpoint record (that was written when the archive began)

The effective date and time of the archive equals the date and time of the checkpoint that this archive took as its starting point. This date and time could differ markedly from the time when the archive process was started.

For example, when no one accessed Universal Server after Tuesday at 7 p.m., and you create an archive on Wednesday morning, the effective date and time for that archive is Tuesday night, the time of the last checkpoint. In other words, when there has been no activity after the last checkpoint, Universal Server does not perform another checkpoint at the start of the archive.

Details of an Archive

This section explains what Universal Server does during an archive. You do not need to understand this section to perform an archive; Informix provides it only as background information.

The following list describes the steps involved in the process:

- **ontape** connects and requests an archive.
- **ontape** readies the device.
- Universal Server prepares to create an archive.
- Universal Server builds and sends archive data.
- **ontape** writes data.
- **ontape** and Universal Server commit the archive.

ontape Connects and Requests an Archive

The **ontape** program sends a request to Universal Server to archive all dbspaces and blobspaces. The following paragraph discusses the order in which you archive the dbspaces.

You archive root dbspaces first and, when you archive blobspaces, you archive them before any of the dbspaces. The archive blocks the blobpage allocation for each blob until you archive the blobspace. Universal Server releases the block before the end of the rest of the archive. Universal Server knows when it archives the blobspace because it keeps track of what it archives.

Once **ontape** successfully connects with Universal Server and initiates the archive request, it reads archive backup data, that Universal Server generates, until no more archive data exists for the archive.

ontape Readies the Device

The **ontape** program prompts you to mount a tape on the tape device specified in the configuration file.

Universal Server Prepares to Create an Archive

When Universal Server prepares to create an archive, it performs the following tasks:

- When an event disables a blobspace or dbspace, Universal Server returns an error and aborts the archive.
- Universal Server compares the specified archive level with the information in the archive reserved page.

  When Universal Server cannot find a record of a previous archive on the reserved page, a level-0 archive becomes the only valid archive level. Otherwise, any archive level is valid.

  See the warning in "When the Archive Device is /dev/null" on page 13-28.
- Universal Server temporarily freezes the status of used logical-log files and checks the total amount of free log space. When free space amounts to less than half of one log file, Universal Server refuses the archive request and recommends that you back up the logical-log files.
- Universal Server synchronizes with other archiving processes to guarantee that no two archiving processes simultaneously archive the same dbspace or blobspace.

- Universal Server initiates a checkpoint (called the *archive checkpoint*). The checkpoint marks the beginning of the archive.

  Universal Server uses a timestamp to determine those pages you can archive. Any pages that you create later than the archive checkpoint do not get archived; a page modified after the archive checkpoint has its before-image archived from the physical log rather than from the modified page.

  For example, assume the checkpoint occurs at 3401. (Timestamps are not based on system time.) For a level-0 archive, you must archive all pages containing timestamps less than 3401. As Universal Server reads through disk pages during the archive, pages with timestamps greater than 3401 are ignored. Universal Server relies on the logical-log files to contain records of modifications that occur after 3401.

  The address of the most-recently written record in the current logical-log file is also noted during the checkpoint. This record becomes the last record from the logical-log file that you copy as part of this Universal Server archive.

  Some transactions can occur during an on-line archive procedure. The restore procedure describes how transactions that span the archive tape and the logical-log file get rolled back during a data restore, when necessary.
- Universal Server reads archive history from the archive reserved page, saving the timestamp of the previous archive to set the criteria for determining those pages you must archive in non-level-0 archives.
- When you perform a full-system archive, Universal Server marks all the logical-log files that contain log records from open transactions so that you cannot free them until you archive them.

- Universal Server builds a list of free pages within each chunk that you archive; the unused pages (and the pages devoted to the logical or physical log) do not get archived at the time of the archive checkpoint. This information resides in entries on the dbspace chunk free-list pages and the blobspace free-map page.
- Universal Server creates a temporary table for each dbspace being archived to store before-images from the physical log. These temporary tables reside in the same spaces designated for other temporary tables. (Refer to the *INFORMIX-Universal Server Administrator's Guide* for information on temporary tables.) When Universal Server runs out of space in where it can create the temporary tables, it aborts the archive.

Once Universal Server performs these tasks, it starts an internal archive thread that generates the archive data.

Universal Server Builds and Sends Data

This section describes the order of information Universal Server sends to **ontape**, and the special actions Universal Server takes during the archive to ensure that it archives completely and efficiently.

The Order of the Archive Data

First, Universal Server sends a control page that contains information about the archive o **ontape**. The control page includes the following items:

- List of spaces included in the archive
- Archive level
- Archive timestamp
- Logging information

Following the control page, Universal Server sends data in the following order:

1. Universal Server adds a section including the reserve pages from the root dbspace to the archive.
2. Universal Server adds a section including a snapshot of the logical logs, that contain open transactions at the time of the archive checkpoint, to the archive.

3. When the archive contains any blobspaces, you include them next. Archiving blobspaces early allows blobpage allocation to resume as soon as possible. Universal Server blocks the allocation of blobpages until it archives the blobspace. You archive only the used portion of a blobpage, not the whole page.
4. Following the blobspaces, you archive the dbspaces in no particular order.
5. The introductory tape-control page of each dbspace and blobspace section contains a mapping of the chunks contained within the space being backed up.
6. When Universal Server archives a dbspace or blobspace, it appends the pages from the temporary table used to store before-images from the physical log.
7. The disk-reading portion of the archive procedure completes when Universal Server reaches the last page of the last chunk. Universal Server sends a trailer page to **ontape**, marking the end of the archive data.

How Universal Server Builds Data

Universal Server does not wait for **ontape** to consume archive data before switching to threads that perform other work. Universal Server continues with other processing during an archive.

When you update a page during an archive, the archive process retrieves the before-image of the page from the physical-log file to capture the state of the page at the time of the archive checkpoint. Periodically, Universal Server empties the physical log of pages that it does not need for fast recovery. When this occurs during an archive, Universal Server writes any before-images, that the archive needs, to a temporary table. When Universal Server writes a before image to the archive, Universal Server removes it from the temporary table. See *INFORMIX-Universal Server Administrator's Guide* for a detailed description of physical logging.

Because blobpages do not pass through shared memory, Informix recommends that you to not use the strategy of archiving from the physical log. Universal Server must prevent clients from overwriting blobspace blobpages before you archive them. To accomplish this, Universal Server blocks allocation of blobpages in each blobspace chunk until it archives all used blobpages in the chunk. As soon as you archive the chunk, blobpage allocation in that chunk resumes. This means that during an on-line archive, you cannot insert blobs into a blobspace until you archive the blobspace chunk.

Universal Server does not read mirror chunks for archiving. You archive pages within a mirror chunk only when Universal Server cannot read the page from the primary chunk.

As Universal Server reads each disk page, it applies a set of criteria that determines the disk pages you must archive. Universal Server copies each page that meets the following criteria for archiving, to the archive tape:

- Universal Server did not allocate the page.
- Universal Server uses the list of free pages in chunks that it created at the start of the archive to determine the pages that it allocated.
- The page does not reside in a logical-log file or the physical log.
- This archive level needs the page.

A level-0 archive requires Universal Server to archive all used disk pages containing a timestamp less than the begin-archive checkpoint timestamp.

A level-1 archive directs Universal Server to archive all disk pages containing a timestamp that is less than the archive checkpoint timestamp but greater than the timestamp associated with the most-recent level-0 archive.

A level-2 archive directs Universal Server to archive all disk pages containing a timestamp that is less than the archive checkpoint timestamp but greater than the timestamp associated with the most recent level-1 archive.

ontape Writes Archive Data

Before receiving archive data, **ontape** writes a tape header page to the archive device. The tape header page contains the following information:

- The tape device block size (TAPEBLK)
- The size of tape (TAPESIZE)
- A flag that indicates the tape is for an archive
- A timestamp that indicates the date and time of the archive
- The archive level
- The id number of the logical-log file that contains the checkpoint record that began the archive
- The physical location of that checkpoint record in the logical-log file

When the Archive Device is /dev/null

When the archive device (TAPEDEV) is defined as **/dev/null**, **ontape** does not ask Universal Server to write a page to the device. Instead, **ontape** asks Universal Server to update the active PAGE_ARCH reserved page with the same information that it writes to the header page. (Refer to the preceding list.) Universal Server also copies the checkpoint information to the active PAGE_CKPT reserved page.

With this action, the root dbspace reserved pages receive acknowledgment that an archive has occurred. This event enables Universal Server to make use of newly added or changed resources.

Warning: *A level-0 archive to* ***/dev/null*** *registers as a valid archive. Universal Server permits you to create a level-1 archive on a tape device even when your only level-0 archive was created when the archive device was* ***/dev/null****. Generally, you must avoid this practice because you lack a source that you can use to restore your system.*

Informix recommends that you set the archive device to /**dev**/**null** as convenient practice, however, when you set up your Universal Server database server—moving logs around, adding dbspaces and loading tables. When Universal Server is ready for normal processing, however, you must set TAPEDEV to a proper device.

ontape and Universal Server Commit the Archive

When **ontape** has received all the archive backup data, it notifies Universal Server whether you must commit or abort the archive. Committing a level-0 archive backup has the following implications:

- Newly mirrored dbspaces and blobspaces become available.
- When the archive includes the root dbspace, newly added log files become available.
- The archive becomes available for use during a restore.
- After a level 0 archive, for any pending database-logging changes, the time of the database-logging change request is compared to the last level-0 archive for each dbspace and blobspace that makes up the database. When all the dbspaces and blobspaces were archived after the database-logging change request was made, the logging change takes effect and Universal Server grants database access.

When Universal Server commits the archive, it stores the history of the archive in the archive reserve pages. You use this information with subsequent archive increments.

When the archive ends, Universal Server drops the temporary tables used for physical log pages.

Backing Up Logical-Log Files

You must only use **ontape** to back up logical-log files when you use **ontape** to make your archive tapes.

In addition to backing up logical-log files, you can use **ontape** to accomplish other tasks involved in maintaining and administrating the logical log. For example, you could switch to the next log file, move logical-log files to other dbspaces, or change the size of the logical log. Instructions for those tasks appear in the *INFORMIX-Universal Server Administrator's Guide*.

See "What Is a Logical-Log Backup?" on page 1-4 for a description of a logical-log backup.

Before You Back Up the Logical-Log Files

Before you back up the logical-log files, you need to understand the following issues:

- Whether you need to back up the logical-log files
- When you need to back up the logical-log files

When you decide you need to back up the logical-log files, you must decide the type of backup you want to perform—automatic or continuous.

Do You Need to Back Up the Logical-Log Files?

When you specify logging for your databases, Universal Server records transactions that occur between archives in the *logical log*, and that consists of a finite number of *logical-log files* on disk. Universal Server continually needs to write new log records but also retain the log records it has already written in case you must restore those transactions. To retain the records in the logical log, yet allow Universal Server to continue writing new log records in a finite amount of space, you must free full log files. To do this, copy them to a safe place on disk or tape.

When You Do Not Use Logging

Remember, even when you do not use logging for any of your databases, you can still perform log backups. These backups can be very small because they contain only administrative information such as checkpoint records and additions and deletions of chunks. When you back up these logical-log files, you can do warm restores even when you do not use logging for any of your databases.

Blobspace Blobs and Logical-Log Files

You must keep the following two points in mind when you use blob data in a database that uses transaction logging:

- To ensure timely reuse of blobpages, you need to back up logical-log files. When users delete blobs in blobspaces, the blobpages do not become freed for reuse until you free the log file containing the delete records. To free the log file, you must back it up.
- When you must back up an unavailable blobspace, **ontape** skips it, making it impossible to recover the blob when that must become necessary. (However, blobpages from deleted blobs do become free when the blobspace becomes available, even though the blob was not backed up.)

In addition, regardless of whether the database uses transaction logging, when you create a blobspace or add a chunk to a blobspace, the blobspace or new chunk is not available for use until the log file that records the event is not the current log file. See the *INFORMIX-Universal Server Administrator's Guide* for information on switching log files.

When You Do Not Need to Recover

When you decide that you do not need to recover transactions or administrative database activities between archives, you can set the Universal Server configuration parameter LTAPEDEV to **/dev/null**.

Warning: *When you set LTAPEDEV to **/dev/null**, it has the following implications:*

- *You can only restore the data, that your Universal Server database server manages, up to the point of your most-recent archive and any previously backed-up logical-log files.*
- *When you recover, you must always perform a full-system restore. (See "A Full-System Restore" on page 13-40.) You cannot perform partial restores or restore when Universal Server is in on-line mode.*

When you set LTAPEDEV to **/dev/null**, Universal Server marks a logical-log file as backed up (status B) as soon as it becomes full. When you close the last open transaction in the log, Universal Server marks the log file free (status F). Universal Server can then reuse that log file without waiting for you to back it up. As a result, Universal Server does not preserve any logical-log records.

Other Universal Server mechanisms that use the logical log, like fast recovery and rolling back transactions, are not impaired when you use **/dev/null** as your log-file backup device. See the *INFORMIX-Universal Server Administrator's Guide* for a description of Universal Server fast recovery. See the ROLLBACK WORK statement in the *Informix Guide to SQL: Syntax* for information about rolling back transactions.

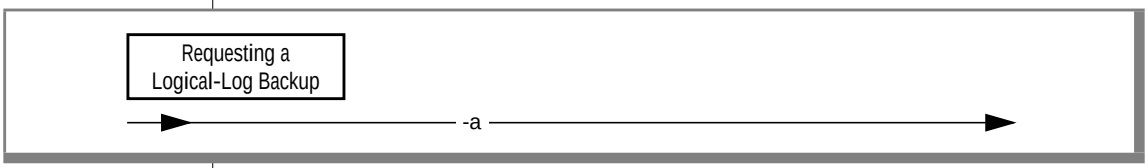
When Must You Back Up Logical-Log Files?

You must attempt to back up each logical-log file as soon as it fills. You can tell when you can back up a log file because it has a *used* status. For more information on monitoring the status of logical-log files, see the *INFORMIX-Universal Server Administrator's Guide*.

Starting an Automatic Logical-Log Backup

Universal Server can operate in on-line mode when you back up logical-log files.

To back up the logical-log files, use the **-a** option of the **ontape** command.



-a directs **ontape** to back up all full logical-log files.

The **-a** option backs up all full logical-log files and prompts you with an option to switch the log files and back up the formerly *current* log.

When the tape mounted on LTAPEDEV becomes full before the end of the logical-log file, **ontape** prompts you to mount a new tape.

When you press the Interrupt key while a backup occurs, Universal Server finishes the backup and then returns control to you. Any other full log files receive a used status.

Example

To back up all full logical-log files, execute the following command:

```
% ontape -a
```

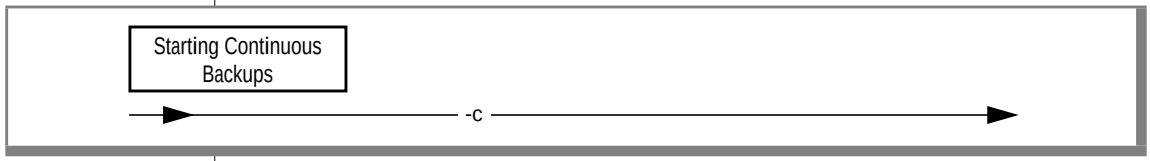
Starting a Continuous Logical-Log File Backup

When you do not want to monitor the log files and start backups when the log files become full, you can start a continuous backup.

When you start a continuous backup, Universal Server automatically backs up each logical-log file as it becomes full. When you perform continuous logical-log file backups, Universal Server protects you against ever losing more than a partial log file, even in the worst-case media failure when a chunk containing logical-log files fails.

With continuous backups you also do not need to remember to back up the log files, but someone must always make media available for the backup process. Also, you must dedicate the backup device and a terminal to the backup process.

To start a continuous backup of the logical-log files, use the **-c** option of the **ontape** command.



-c directs **ontape** to initiate continuous backup of logical-log files.

The **-c** option initiates continuous logging. Universal Server backs up each logical-log file as it becomes full. Continuous backup does not back up the current log file.

Universal Server can operate in on-line mode when you start continuous backups. Execute the following command to start continuous logging:

```
% ontape -c
```

See "Ending Continuous Logical-Log File Backup".

When the tape mounted on LTAPEDEV becomes full before the end of the logical-log file, Universal Server prompts the operator for a new tape.

Ending Continuous Logical-Log File Backup

To end continuous logical-log file backup, press the Interrupt key (CTRL-C).

When you press the Interrupt key while Universal Server backs up a logical-log file to a local device, all logs that were completely backed up before the interrupt get captured on the tape and get marked as backed up by Universal Server.

When you press the Interrupt key while Universal Server waits for a log file to fill (and thus is not backing up any logical-log files), all logs that were backed up before the interrupt reside on the tape and get marked as backed up by Universal Server.

When you press the Interrupt key while Universal Server performs a continuous backup to a remote device, any log files that were backed up during this operation can or cannot reside on the tape, and do not get marked as backed up by Universal Server (a good reason why you must not do continuous remote backups).

After you stop continuous logging, you must start a new tape for subsequent log backup operations.

You must explicitly request logical-log backups (using **ontape -a**) until you restart continuous logging.

What Device Must Logical-Log Backups Use?

The **ontape** utility uses parameters defined in the ONCONFIG file to define the tape device for logical-log backups, as explained in Chapter 12, "Configuring ontape." However, consider the following issues when you choose a logical-log backup device:

- When the logical-log device differs from the archive device, you can plan your backups without considering the competing needs of the archive schedule.
- When you specify **/dev/null** as the logical-log backup device in the ONCONFIG parameter LTAPEDEV, you avoid having to mount and maintain backup tapes. However, you can only recover Universal Server data up to the point of your most-recent archive tape. You cannot restore work done after the archive. See the warning about setting LTAPEDEV to **/dev/null** in "When You Do Not Need to Recover" on page 13-31.
- When your tape device runs slow, the logical log could fill up faster than you can copy it to tape. In this case, you could consider performing the backup to disk and then copying the disk backup to tape.

Details of a Logical-Log File Backup

This section describes the steps that **ontape** and Universal Server perform while backing up a logical-log file to tape. You do not need to understand this section to back up logical-log files. It exists for your information only.

The following list describes these steps:

1. **ontape** connects and requests a log-file backup.
2. **ontape** readies the device and tape.
3. Universal Server prepares to back up log data.
4. Universal Server builds and sends logical-log file data.
5. **ontape** writes the logical-log backup data.
6. **ontape** and Universal Server commit the backup.

The following sections explain each step in detail.

ontape Connects and Requests a Backup

The **ontape** program connects to Universal Server and sends a request to back up the logical-log files.

ontape Readies the Device and Tape

The **ontape** program prompts you to mount a tape on the tape device specified in the configuration file.

When you use a new tape, **ontape** writes a tape header (also called a volume header) to the device.

Universal Server Prepares to Back Up a Logical-Log File

When Universal Server receives a request for a log-file backup, it locates the oldest logical-log file that has been used but not backed up (status U when you run **onstat -l**). Universal Server also checks to see that no other log backups are occurring.

Next, Universal Server starts an internal thread that collects the log-file data and sends it to **ontape**.

Universal Server Builds and Sends Logical-Log Data

Universal Server builds the data that needs backing up and sends it to **ontape**. The **ontape** program writes the data to the backup device.

Universal Server Sends Blobpages

Universal Server begins by comparing the identification number of the log file it backs up with every blobspace blob. (It actually looks at every blobspace free-map page). Universal Server looks for blobpages that were allocated or marked for deletion during the time this logical-log file was the current log file.

When blobpages exist for copying, each blobpage that was allocated or marked for deletion during the time that this log file was current gets sent to the client. Universal Server precedes each blobpage with a blob header and follows it with a blob trailer.

When a blobspace containing blobpages that needs backing up is unavailable at the time a backup (or salvaging) of the log files occurs, Universal Server does not wait for the blobspace to become available. It continues the log-file backup without copying the blobpages it needs. Thus, you cannot restore the blob when you roll the logical-log file forward, and you lose the blob during a restore.

Universal Server Sends a Log Header

After Universal Server checks all blobs and sends the required blobpages to **ontape**, Universal Server creates a log header and sends it to **ontape**.

The log header differs from the tape header. The log header specifies (among other things) the id number of the logical-log file and the number of pages from the logical-log file that you must copy.

Universal Server Sends Log Records

Following the log header, Universal Server begins sending each page in the logical-log file that it backs up. When some pages in the log file go unused (for example, when you back up a file before it fills up), Universal Server does not write the unused pages to tape.

Universal Sever Sends a Log Trailer

After it sends the last page in the log file, Universal Server sends **ontape** a log trailer.

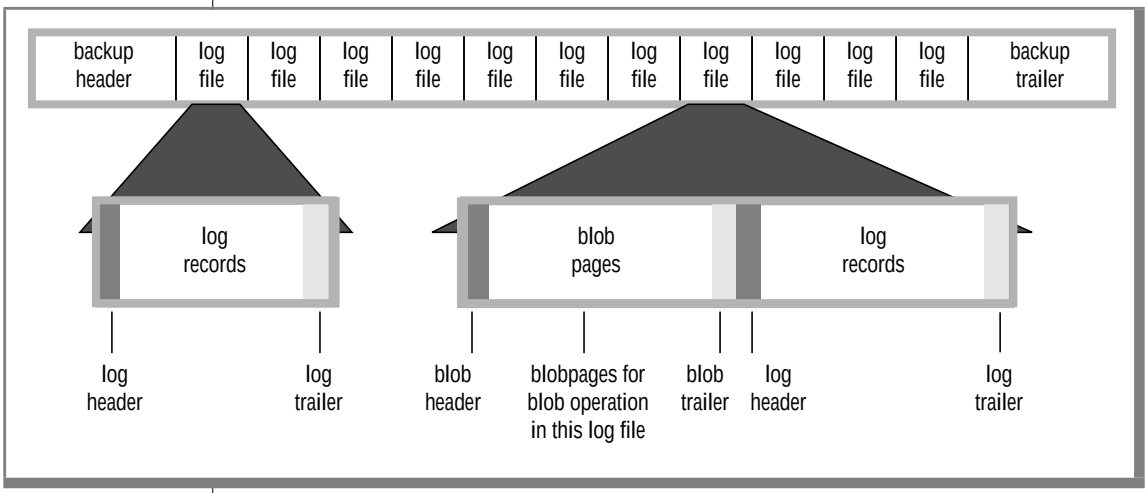
ontape Writes Data to the Backup Device

The **ontape** program writes the logical-log data to the backup device in the same order that it receives it from Universal Server. The following list shows that order:

- Blobpages
- Log-file header
- Log-file records
- Log-file trailer

Figure 13-3 illustrates the order of information on the logical-log backup tape.

Figure 13-3
Logical-Log Backup Tape Format



ontape and Universal Server Commit the Backup

When **ontape** has secured all the log-file backup data, it notifies Universal Server whether you must commit or abort the log-file backup. Committing the log-file backup changes the status to backed-up. Aborting the log-file backup leaves the log file in the same state as it was prior to the log-file backup. When it changes the log-file status, Universal Server checks to see when you can free the log file for reuse.

ontape Looks for More Log Files to Back Up

After you successfully back up a logical-log file, **ontape** determines whether you must back up another log file. When no other log files exist to back up and the user specified that Universal Server must back up the current file log, Universal Server switches the current log file to the next log file and backs up the formerly current log file. When the log backup continues, it repeats the steps outlined in the preceding three sections: "Universal Server Builds and Sends Logical-Log Data" on page 13-36, "ontape Writes Data to the Backup Device" on page 13-37, and "ontape and Universal Server Commit the Backup".

When no more logs exist to back up and **ontape** performs a continuous log backup, **ontape** goes into a loop where it waits awhile and then again asks Universal Server if you can find full log files to back up. As the log files fill, Universal Server backs them up.

ontape Writes the Trailer Page

When the entire log-backup process ends, **ontape** writes a backup trailer to indicate the end of the backup session.

When You Need a New Tape

When you need more than one tape during the logical-log backup, **ontape** provides you with labelling information for the full tape and prompts you to mount a new tape.

Restoring Universal Data

This section provides instructions for restoring data using **ontape**. It provides instructions for the following procedures:

- A full-system restore
- A restore of selected dbspaces or blobspaces

Before you start restoring data, you must understand the concepts in "What Is a Universal Server Restore?" on page 1-6. As explained in that section, a complete recovery of Universal Server data generally consists of a physical restore and a logical restore.

Choosing the Type of Physical Restore

Before you restore Universal Server data spaces due to a failure that *caused* Universal Server *to go to off-line mode*, you must restore all the data that Universal Server manages. You call this type of restore a *full-system* restore. When the failure did not cause Universal Server to go to off-line mode, you can restore only selected dbspaces and blobspaces—that is, only those that the failure affected.

A Full-System Restore

When your Universal Server database server goes to off-line mode because of a disk failure or corrupted data, it means that a *critical dbspace* was damaged. The following list shows critical dbspaces:

- The root dbspace
- The dbspace containing the physical log
- A dbspace containing logical-log files

When you need to restore any critical dbspace, you must perform a full system restore to restore all the data that your Universal Server database server manages. You must start a full-system restore with a *cold restore*. See "Choosing a Universal Server Mode—Cold, Warm, or Mixed Restore".

Restoring Selected Dbspaces and Blobspaces

When your Universal Server database server *does not* go to off-line mode because of a disk failure or corrupted data, the damage occurred to a noncritical dbspace or blobspace.

When you do not need to restore a critical dbspace, you can restore only those dbspaces and blobspaces that contain a damaged chunk or chunks. When a a media failure occurs in one chunk of a dbspace or blobspace that spans multiple chunks, all active transactions for that dbspace or blobspace must terminate before Universal Server can restore it. You can start a restore operation before Universal Server finishes the transactions, but the restore becomes delayed until Universal Server verifies that you finished all transactions that were active at the time of the failure.

Choosing a Universal Server Mode—Cold, Warm, or Mixed Restore

When you restore Universal Server data, you must decide whether you can do it while Universal Server operates in off-line mode or on-line mode. This decision, while not completely arbitrary, depends in part on the data you intend to restore. The following sections explain the factors that determine the Universal Server mode you must use when you perform a restore.

A Cold Restore

You perform a *cold restore* while Universal Server operates in off-line mode. It consists of both a physical restore and a logical restore. You must perform a cold restore to restore any critical dbspaces.

As shown in Figure 13-4, you can restore all the dbspaces and blobspaces that Universal Server manages (a full-system restore) with a complete cold restore.

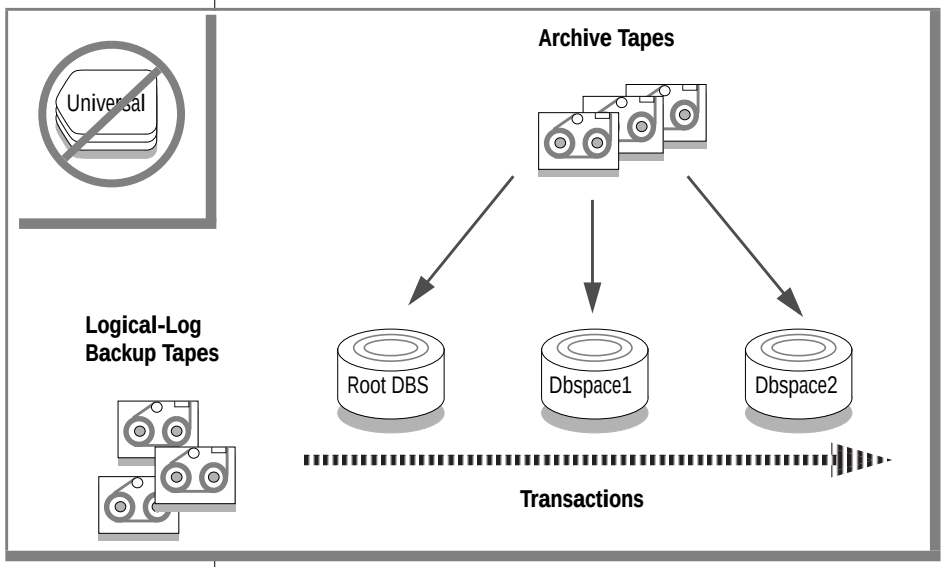


Figure 13-4
A Full-System Cold Restore

Universal Server is off-line when you begin a cold restore but it goes into recovery mode after it restores the reserved pages. From that point on it stays in recovery mode until either a logical restore finishes (after which it works in quiescent mode) or you use the **onmode** utility to shift it to another mode.

A Warm Restore

A warm restore restores noncritical dbspaces and blobspaces while Universal Server is in on-line or quiescent mode. It consists of one or more physical restore operations (when you restore multiple dbspaces or blobspaces concurrently), a logical-log backup, and a logical restore. Figure 13-5 depicts a warm restore.

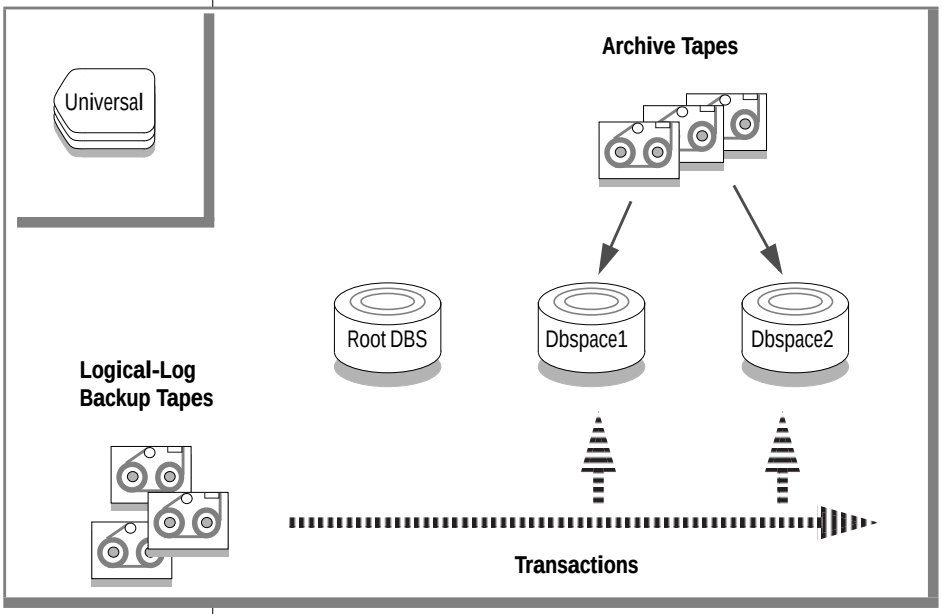


Figure 13-5
A Warm Restore

During a warm restore, Universal Server replays backed-up logical-log files for the dbspaces you restore. To avoid overwriting the current logical log, Universal Server writes the logical-log files, that you designate for replay, to temporary space. Therefore, a warm restore requires enough temporary space to hold the logical log (one set of logical-log files) or the number of log files being replayed, whichever is smaller. See the discussion of **DBSPACETEMP** in the *INFORMIX-Universal Server Administrator's Guide* for information on how INFORMIX-Universal Server looks for temporary space.

Warning: *Make sure enough temporary space exists for the logical-log portion of the warm restore; the maximum amount of temporary space that Universal Server needs equals the size of the logical log (the size of all the logical-log files).*

A Mixed Restore

A *mixed restore* is a cold restore followed by a warm restore. A mixed restore restores some dbspaces and blobspaces during a cold restore (Universal Server is off-line) and some dbspaces and blobspaces during a warm restore (Universal Server is on-line). You could do a mixed restore when you perform a full-system restore, but you need to provide access to a particular table or set of tables as soon as possible. In this case, you perform a cold restore to restore the critical dbspaces and the dbspaces containing the important tables. Figure 13-6 illustrates the cold portion of a mixed restore.

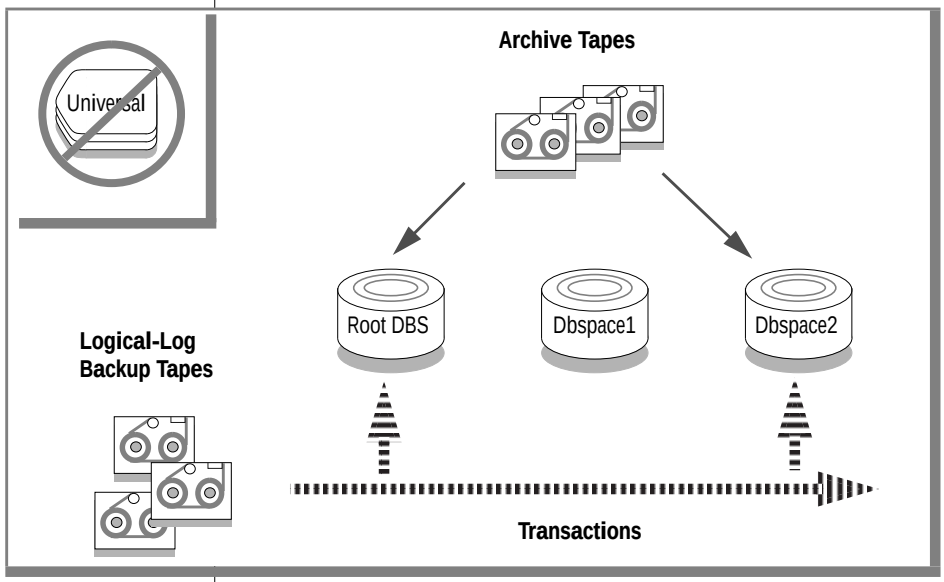


Figure 13-6
The Cold Portion of a Mixed Restore

Following the cold restore, you place Universal Server in on-line mode and perform a warm restore to restore the remaining dbspaces. Figure 13-7 illustrates the warm portion of a mixed restore.

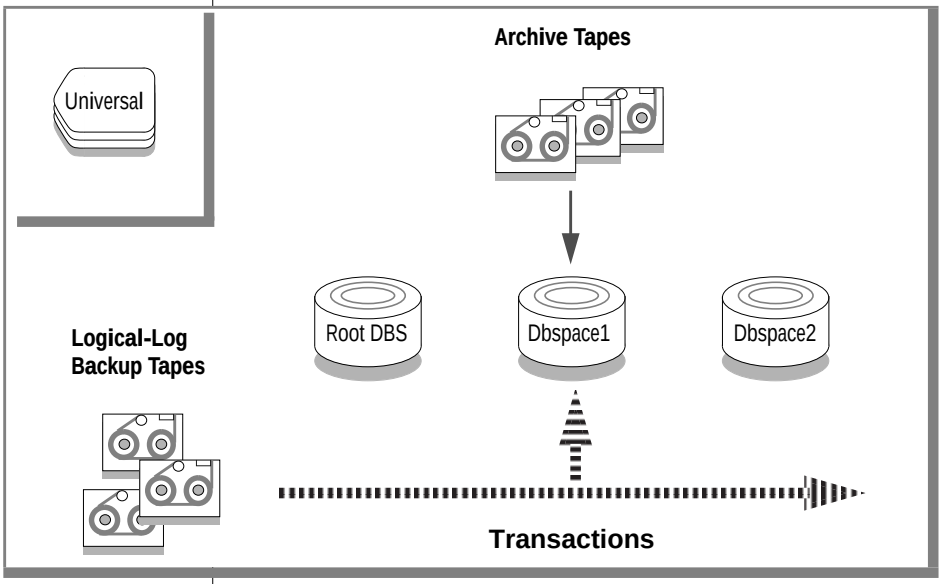


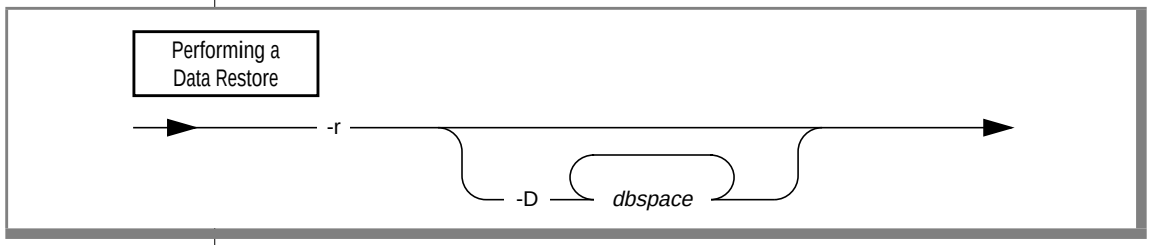
Figure 13-7
The Warm Portion of a Mixed Restore

A cold restore takes less total time to restore all your Universal Server data than a mixed restore, even though Universal Server is on-line during part of a mixed restore because a mixed restore requires two logical restores (one for the cold restore and one for the warm restore). A mixed restore, however, requires Universal Server to go *off-line* for less time than a cold restore.

The dbspaces not restored during the cold restore do not become available until after Universal Server restores them during a warm restore, even though a critical dbspace possibly did not damage them.

Performing a Restore

You use the **-r ontape** option to perform a full physical and logical restore of Universal Server data. You use the -D **ontape** option to restore selected Universal Server dbspaces and blobspaces.



-r directs **ontape** to perform a data restore (both physical and logical).

-D directs **ontape** to restore only the dbspaces or blobspaces you specify as part of a warm restore. Universal Server must go into on-line or quiescent mode to do a warm restore.

dbspace is the name of a dbspace or blobspace to restore.

When you do not specify the **-D** option, **ontape** performs a full-system restore. Universal Server must go off-line to do a full-system restore.

The **-r** option restores Universal Server data from the archive tape and the logical log backup tapes you created after (and including) your last level-0 archive. When you use the **-D** option, you can restore selected dbspaces or blobspaces. See "Steps to Restore Selected Dbspaces" on page 13-50 for information.

Steps to Restore the Whole System

This section outlines the steps you need to perform to restore your entire Universal Server database server with **ontape**. The following list describes the main steps in a full-system restore:

1. Gather the appropriate tapes.
2. Decide on a complete cold or a mixed restore.
3. Verify your Universal Server configuration.
4. Perform a cold restore.

Read these instructions familiarize yourself with them before you attempt a full-system restore.

Gather the Appropriate Tapes

You must gather the appropriate archive and logical-log tapes.

Archive Tapes

Before you start your restore, gather together all the tapes from your latest level-0 archive containing the dbspaces and blobspaces you are restoring, and any subsequent level-1 or level-2 archives.

Identify the tape that has the latest level-0 archive of the root dbspace on it; you must use this tape first.

Logical-Log Tapes

Gather together all the logical-log tapes from the backup prior to the latest level-0 archive of the dbspaces and blobspaces you are restoring.

Decide on a Complete Cold or a Mixed Restore

As mentioned in "Choosing a Universal Server Mode—Cold, Warm, or Mixed Restore" on page 13-40, when you restore your entire Universal Server database server, you can restore the critical dbspaces (and any other dbspaces or blobspaces you want to come on-line quickly) during a cold restore, and then restore the remaining dbspaces and blobspaces during a warm restore. Decide before you start the restore if you want a completely cold, or mixed, restore.

Verify Your Universal Server Configuration

During a cold restore, you cannot reinitialize shared memory, add chunks, or change tape devices. Thus, when you begin the restore, the current Universal Server configuration must remain compatible with, and accommodate, all parameter values assigned after the time of the most-recent archive.

For guidance, use the copies of the configuration file that you create at the time of each archive. However, do not blindly set all current parameters to the same values as were recorded at the last archive. Pay attention to the following three groups of parameters:

- Shared-memory parameters
- Mirroring parameters
- Device parameters

Set Shared-Memory Parameters to Maximum Assigned Value

Make sure that you set your current shared-memory parameters to the *maximum* value assigned after the level-0 archive. For example, When you decreased the value of USERTHREADS from 45 to 30 sometime after the level-0 archive, you must begin the restore with USERTHREADS set at 45, and not at 30, even though the configuration file copy for the last archive could register the value of USERTHREADS set at 30. (When you do not possess a record of the maximum value of USERTHREADS after the level-0 archive, set the value as high as you think necessary. You could reassign values to BUFFERS, LOCKS, and TBLSPACES as well, because the minimum values for these three parameters are based on the value of USERTHREADS.)

Set Mirroring Configuration to Level-0 Archive State

Verify that your current mirroring configuration matches the configuration that was in effect at the time of the last level-0 archive. Because Informix recommends that you create a level-0 archive after each change in your mirroring configuration, this creates no problems. The most critical parameters are the mirroring parameters that appear in the Universal Server configuration file, MIRRORPATH and MIRROROFFSET.

Ensure That Needed Devices Are Available

Verify that the raw devices or files that you used for Universal Server storage (of the dbspaces and blobspaces being restored) after the level-0 archive are available.

For example, when you dropped a dbspace or mirroring for a dbspace after your level-0 archive, you must make the dbspace or mirror chunk device available to Universal Server when you begin the restore. When Universal Server attempts to write to the chunk and cannot find it, the restore does not complete. Similarly, when you added a chunk after your last archive, you must make the chunk device available to Universal Server when it begins to roll forward the logical logs.

Perform a Cold Restore

To perform a cold restore, Universal Server must operate in off-line mode.

You must log in as **informix** or **root** to use **ontape**. Execute the following **ontape** command to restore all the dbspaces and blobspaces that Universal Server manages:

```
% ontape -r
```

When you perform a mixed restore, you restore only some of the dbspaces or blobspaces that Universal Server manages during the cold restore. You must restore at least all the critical dbspaces (the root dbspace, and dbspaces with the physical-log and logical-log files), as shown in the following example:

```
% ontape -r -D rootdbs llogdbs plogdbs
```

Salvage Logical-Log Files

Before the restore starts, the console prompts you to salvage the logical-log files on disk. Consider yourself wise when you salvage the log files. It saves log records that you did not back up and enables you to recover your Universal Server data up to the point of the failure.

Use a new tape to salvage the log files.

Mount Tapes During the Restore

During the restore, **ontape** prompts you to mount tapes with the appropriate dbspaces or log files.

Restore Logical-Log Files

When you perform a mixed restore, you must restore all the logical-log files backed up after the last level-0 archive.

When you perform a full restore, you can choose not to restore logical-log files. When you do not back up your logical-log files, or choose not to restore them, you can restore your Universal Server data only up to the state it was in at the time of your last archive. See "Do You Need to Back Up the Logical-Log Files?" on page 13-30 for more information.

When the Restore Is Over

At the end of the cold restore, Universal Server is in quiescent mode. You can bring Universal Server into on-line mode at this point and continue processing as usual.

When you only restored some of your dbspaces and blobspaces during the cold restore, you can start a warm restore of the remaining dbspaces and blobspaces after you bring Universal Server into on-line mode.

Steps to Restore Selected Dbspaces

This section outlines the steps you must perform during a restore of selected dbspaces or blobspaces with **ontape** while Universal Server is in on-line or quiescent mode (a warm restore). The following list describes the main steps in a warm restore:

1. Gather the appropriate tapes.
2. Verify your Universal Server configuration.
3. Back up logical-log files.
4. Perform a warm restore.

Read these instructions and familiarize yourself with them before you attempt a restore.

Gather the Appropriate Tapes

You must gather the appropriate archive and logical-log tapes.

Archive Tapes

Before you start your restore, gather together all the tapes from your latest level-0 archive containing the dbspaces and blobspaces you are restoring, and any subsequent level-1 or level-2 archives.

Logical-Log Tapes

Gather together all the logical-log tapes from the logical-log backup prior to the latest level-0 archive of the dbspaces and blobspaces you are restoring.

Verify Your Universal Server Configuration

During a warm restore, you do not need to worry about shared-memory parameters, as you do for cold restores.

Ensure That Needed Devices Are Available

Verify that you can use the raw devices or files that you used for Universal Server storage (of the dbspaces and blobspaces being restored) after the level-0 archive.

For example, when you dropped a dbspace or mirroring for a dbspace after your level-0 archive, you must make the dbspace or mirror chunk device available to Universal Server when you begin the restore. When Universal Server attempts to write to the chunk and cannot find the chunk, the restore does not complete. Similarly, when you added a chunk after your last archive, you must make the chunk device available to Universal Server when it begins to roll forward the logical logs.

Back Up Logical-Log Files

Before you start a warm restore (even when you perform the warm restore as part of a mixed restore), you must back up your logical-log files. See "Backing Up Logical-Log Files" on page 13-29.

After the warm restore, you *must* roll forward your logical-log files to bring the dbspaces that you are restoring to a state of consistency with the other dbspaces in the system. Failure to roll forward the logical log after restoring a selected dbspace results in the following message from **ontape**:

```
Partial system restore is incomplete.
```

Perform a Warm Restore

To perform a warm restore, Universal Server must operate in on-line or quiescent mode.

You must log in as **informix** or **root** to use **ontape**. Execute the **ontape** command, with the options shown in the following example, to restore selected dbspaces and blobspaces that Universal Server manages:

```
% ontape -r -D dbspace1 dbspace2
```

You cannot restore critical dbspaces (the root dbspace, and dbspaces with the physical log and logical-log files) during a warm restore; you must restore them as part of a cold restore, described in "Steps to Restore the Whole System" on page 13-46.

During the restore, **ontape** prompts you to mount tapes with the appropriate dbspaces or log files.

At the end of the warm restore, the dbspaces or blobspaces that were down go on-line.

Index

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X