

IBM Informix Backup and Restore Guide

IBM Informix Extended Parallel Server, Version 8.4
IBM Informix Dynamic Server, Version 9.4

March 2003
Part No. CT1SLNA

Note:

Before using this information and the product it supports, read the information in the appendix entitled "Notices."

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties, and any statements provided in this manual should not be interpreted as such.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1996, 2003. All rights reserved.

US Government User Restricted Rights—Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Table of Contents

Introduction

In This Introduction	3
About This Manual	3
Types of Users	3
Software Dependencies	4
Assumptions About Your Locale.	4
Demonstration Database	5
New Features in Dynamic Server, Version 9.4	6
Features from Dynamic Server 9.3	6
Features from Dynamic Server 9.21	7
Organizational Changes to This Manual Since Version 9.2	7
New Features in Extended Parallel Server	8
Documentation Conventions	8
Typographical Conventions	8
Icon Conventions	9
Command-Line Conventions	10
Additional Documentation	13
Related Reading	15
Compliance with Industry Standards	16
IBM Welcomes Your Comments	16

Section I Introducing Backup and Restore

Chapter 1 **Introducing Backup and Restore**

In This Chapter	1-3
What Is a Recovery System?	1-3
What Is a Backup?	1-4
What Is a Logical-Log Backup?	1-5
What Is a Restore?	1-8
Comparing ON-Bar and ontape	1-12
Planning a Recovery Strategy	1-14
What Types of Data Loss Can Occur?	1-14
How Severe is the Failure?	1-15
How Do You Use Your Data?	1-16
Scheduling Backups.	1-17
Planning a Backup System for a Production Database Server	1-18
Evaluating Hardware and Memory Resources.	1-18
Evaluating Backup and Restore Time	1-18
Evaluating Logging and Transaction Activity	1-20

Section II The ON-Bar Backup and Restore System

Chapter 2 **The ON-Bar Backup and Restore System**

In This Chapter	2-3
Where to Find Information on ON-Bar and ISM	2-4
ON-Bar for Dynamic Server	2-6
ON-Bar for Extended Parallel Server	2-9
Database Server and Storage-Manager Communication	2-9
Backup Scheduler	2-10
ON-Bar Utilities	2-12
IBM Informix Storage Manager	2-13
Third-Party Storage Managers	2-13
The XBSA Interface	2-14
The ON-Bar Tables	2-15
The Emergency Boot Files.	2-15
The ON-Bar Activity Log	2-17

Chapter 3

Configuring the Storage Manager and ON-Bar

In This Chapter	3-3
Configuring a Storage Manager	3-3
Installing and Configuring a Third-Party Storage Manager	3-3
Configuring ISM	3-4
Updating the sm_versions File	3-5
Configuring Multiple Storage Managers on Coserver Nodes	3-6
Configuring ON-Bar	3-7
Creating the bargroup Group	3-7
Updating the onbar Script	3-8
Setting ISM Environment Variables and ONCONFIG Parameters	3-9
Specifying the Location of the XBSA Library	3-9
Using ON-Bar Configuration Parameters	3-11
Using ON-Bar Configuration Parameters on Dynamic Server	3-11
Using ON-Bar Configuration Parameters on Extended Parallel Server	3-12
Before You Make a Test Backup	3-19
Choosing Storage Managers and Storage Devices	3-20
Features That ISM Supports	3-20
Features That ISM Does Not Support	3-21
Storage Device Requirements	3-22
Considerations for Extended Parallel Server	3-23

Chapter 4

Backing Up with ON-Bar

In This Chapter	4-3
Syntax of ON-Bar Commands	4-4
Preparing for a Backup	4-5
What Data Does ON-Bar Back Up?	4-5
Which Administrative Files to Back Up?	4-6
Installing and Configuring a Storage Manager	4-7
What Is a Whole-System Backup?	4-7
What Is a Standard Backup?	4-7
What Is a Physical Backup?.	4-7
Choosing a Backup Level	4-8
Collecting Information About Your System Before a Backup	4-9
Backing Up Storage Spaces and Logical Logs	4-10
Backup Syntax	4-11
Backing Up After Changing the Physical Schema	4-13
Using ISM During a Backup	4-14

Using IBM Informix Server Administrator to	
Back Up and Verify	4-15
ON-Bar Backup Examples.	4-16
Backing Up Table Types	4-21
Backing Up Logical Logs	4-21
Backing Up Logical Logs on Dynamic Server	4-22
Backing Up Logical Logs on Extended Parallel Server	4-25
Monitoring Logical-Log Backups	4-27
Salvaging Logical-Log Files	4-28
Skipping Logical Replay	4-29
Restoring Nonlogging Databases and Tables	4-29
Understanding ON-Bar Backup Processes	4-30
Backup Sequence on Dynamic Server.	4-30
Backup Sequence on Extended Parallel Server.	4-32

Chapter 5 **Verifying Backups**

In This Chapter	5-3
Verifying Backups with archecker	5-3
Using archecker to Verify Backups.	5-4
Preparing to Verify Backups	5-6
Syntax	5-7
Estimating the Amount of Temporary Space for archecker	5-8
Verification Examples	5-10
Interpreting Verification Messages.	5-11
What To Do If Backup Verification Fails	5-12
Procedures for Fixing Backup Verification Problems.	5-13

Chapter 6 **Restoring Data with ON-Bar**

In This Chapter	6-3
What Types of Restores Does ON-Bar Perform	6-3
What Is a Warm Restore?	6-3
What Is a Cold Restore?	6-4
What Is a Mixed Restore?	6-5
What Is a Parallel Restore?	6-5
What Is a Whole-System Restore?	6-5
What Is a Point-in-Time Restore?	6-6
What Is an Imported Restore?	6-6
What is a Rename Chunks Restore?	6-6
What Is a Restartable Restore?	6-7

Using the Pre-Recovery Checklist	6-7
Monitoring Restores	6-8
Ensuring That Storage Devices Are Available	6-9
Restoring Save Sets with ISM	6-10
Performing a Complete Restore	6-10
Performing a Physical-Only or Logical-Only Restore	6-13
Using IBM Informix Server Administrator to Restore Data	6-15
Examples of ON-Bar Restore Commands	6-16
Renaming Chunks During a Restore	6-27
Key Considerations	6-27
New-Chunk Requirements	6-28
Syntax	6-28
Examples of Renaming Chunks During a Restore	6-30
Transferring Data with the Imported Restore	6-32
Importing a Restore	6-34
Initializing High-Availability Data Replication with ON-Bar	6-36
Restoring Table Types	6-38
Using Restartable Restore to Recover Data	6-39
Restartable Restore Example	6-40
Restarting a Restore	6-41
Resolving a Failed Restore	6-44
Understanding ON-Bar Restore Processes	6-46
Warm-Restore Sequence on Dynamic Server	6-46
Cold-Restore Sequence on Dynamic Server	6-48
Warm-Restore Sequence on Extended Parallel Server	6-50
Cold-Restore Sequence on Extended Parallel Server	6-52

Chapter 7 Performing an External Backup and Restore

In This Chapter	7-3
Recovering Data Using an External Backup and Restore	7-3
What Is Backed Up in an External Backup?	7-5
Rules for an External Backup	7-5
Performing an External Backup	7-7
Preparing for an External Backup	7-9
Blocking and Unblocking Dynamic Server	7-9
Blocking and Unblocking Extended Parallel Server	7-10
Monitoring an External Backup	7-15
Tracking an External Backup	7-15

What Is Restored in an External Restore?	7-16
Using External Restore Commands	7-17
Rules for an External Restore.	7-18
Performing an External Restore.	7-19
Initializing HDR with an External Backup and Restore.	7-23

Chapter 8 **Using ONBar Utilities**

In This Chapter	8-3
Customizing ON-Bar and Storage-Manager Commands	8-3
Printing the Backup Boot Files	8-4
Migrating Backed-Up Logical Logs to Tape.	8-6
Using start_worker.sh to Start onbar_worker Processes Manually	8-7
Expiring and Synchronizing the Backup Catalogs	8-8
Choosing an Expiration Policy	8-9
Using the onmsync Utility	8-10
Starting and Stopping ON-Bar Sessions	8-14
Using the onbar_w Utility.	8-15
Monitoring the Backup Scheduler Status	8-17
Using the onstat -g bus Option	8-17
Using the onstat -g bus_sm Option	8-18

Chapter 9 **Setting ON-Bar Configuration Parameters**

In This Chapter	9-3
Setting archecker Configuration Parameters in AC_CONFIG	9-4
AC_CONFIG File	9-4
AC_MSGPATH	9-5
AC_STORAGE	9-5
AC_VERBOSE	9-6
Setting ON-Bar Parameters in ONCONFIG	9-7
ALARMPROGRAM	9-7
BAR_ACT_LOG	9-8
BAR_BOOT_DIR	9-9
BAR_BSALIB_PATH	9-10
BAR_DBS_COSVR	9-11
BAR_HISTORY	9-12
BAR_IDLE_TIMEOUT	9-13
BAR_LOG_COSVR	9-13
BAR_MAX_BACKUP	9-14
BAR_NB_XPORT_COUNT	9-15
BAR_PROGRESS_FREQ	9-16

BAR_RETRY	9-17
BAR_SM	9-19
BAR_SM_NAME	9-19
BAR_WORKER_COSVR	9-20
BAR_WORKER_MAX	9-20
BAR_XFER_BUF_SIZE	9-22
BAR_XFER_BUFSIZE.	9-23
BAR_XPORT_COUNT	9-24
ISM_DATA_POOL.	9-24
ISM_LOG_POOL	9-25
LOG_BACKUP_MODE	9-25
LTAPEDEV	9-26
RESTARTABLE_RESTORE	9-27
Files That ON-Bar and ISM Use	9-28

Chapter 10 **ON-Bar Catalog Tables**

In This Chapter	10-3
The bar_action Table	10-3
The bar_instance Table	10-5
The bar_object Table	10-7
The bar_server Table	10-8
The bar_version Table	10-8
ON-Bar Catalog Map	10-9
Backup Scheduler SMI Tables	10-11
sysbuobject	10-12
sysbuobjses	10-13
sysbusession.	10-14
sysbusm	10-14
sysbusmdbspace	10-15
sysbusmlog	10-15
sysbusmworker.	10-16
sysbuworker.	10-16

Chapter 11	ON-Bar Messages and Return Codes	
	In This Chapter	11-3
	About ON-Bar Messages	11-3
	Message Format	11-3
	Message Numbers	11-4
	ON-Bar Usage Messages	11-5
	ON-Bar Return Codes	11-9

Section III The ontape Backup and Restore System

Chapter 12	Configuring ontape	
	In This Chapter	12-3
	Setting the ontape Configuration Parameters	12-3
	Setting the Tape-Device Parameters	12-4
	Specifying the Tape-Block-Size Parameters	12-7
	Specifying the Tape-Size Parameters	12-7
	Checking and Changing ontape Configuration Parameters	12-8
	Who Can Change ontape Parameters?	12-8
	When Can You Change ontape Parameters?	12-8
	Changing ontape Parameters	12-10

Chapter 13	Backing Up with ontape	
	In This Chapter	13-3
	Syntax of ontape	13-3
	Starting ontape	13-4
	Using ontape Exit Codes	13-4
	Changing Database Logging Status	13-5
	Creating a Backup	13-6
	Choosing a Backup Level	13-6
	Backing Up After Changing the Physical Schema.	13-7
	Preparing for a Backup.	13-7
	Performing a Backup	13-10
	When the Logical-Log Files Fill During a Backup.	13-12
	When a Backup Terminates Prematurely.	13-13
	Monitoring Backup History Using oncheck.	13-13
	Backing Up Logical-Log Files with ontape	13-14
	Before You Back Up the Logical-Log Files	13-14
	When Must You Back Up Logical-Log Files?	13-15

Starting an Automatic Logical-Log Backup	13-16
Starting a Continuous Logical-Log File Backup	13-16
Ending a Continuous Logical-Log Backup	13-17
What Device Must Logical-Log Backups Use?	13-18

Chapter 14 Restoring with ontape

In This Chapter	14-3
Choosing the Type of Physical Restore	14-3
A Full-System Restore	14-3
Restoring Selected Dbspaces, Blobspaces, and Sbspaces	14-4
Choosing a Cold, Warm, or Mixed Restore	14-4
A Cold Restore	14-4
A Warm Restore	14-5
A Mixed Restore	14-5
Performing a Restore	14-6
Restoring the Whole System	14-8
Gathering the Appropriate Tapes.	14-8
Deciding on a Complete Cold or a Mixed Restore	14-9
Verifying Your Database Server Configuration	14-9
Performing a Cold Restore	14-10
Restoring Selected Storage Spaces	14-14
Gathering the Appropriate Tapes.	14-14
Ensuring That Needed Device Are Available.	14-15
Backing Up Logical-Log Files	14-15
Performing a Warm Restore	14-15
Restoring Raw Tables	14-16
Renaming Chunks During a Restore	14-16
Key Considerations	14-16
New Chunk Requirements	14-17
Examples of Renaming Chunks During a Restore	14-18

Appendix A	Troubleshooting
Appendix B	onstat Command Reference
Appendix C	Migration
Appendix D	GLS Support
Appendix E	Notices
	Index

Introduction

In This Introduction	3
About This Manual	3
Types of Users	3
Software Dependencies	4
Assumptions About Your Locale	4
Demonstration Database	5
New Features in Dynamic Server, Version 9.4	6
Features from Dynamic Server 9.3	6
Features from Dynamic Server 9.21	7
Organizational Changes to This Manual Since Version 9.2	7
New Features in Extended Parallel Server	8
Documentation Conventions	8
Typographical Conventions	8
Icon Conventions	9
Comment Icons	9
Feature, Product, and Platform Icons	10
Command-Line Conventions	10
How to Read a Command-Line Diagram	12
Additional Documentation	13
Related Reading	15
Compliance with Industry Standards	16
IBM Welcomes Your Comments	16

In This Introduction

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

About This Manual

This manual describes how to use the ON-Bar and **ontape** utilities to back up and restore database server data. These utilities enable you to recover your databases after data is lost or becomes corrupt due to hardware or software failure or accident.

ON-Bar requires IBM Informix Storage Manager, Version 2.2, or a third-party storage manager to manage the storage devices. The backup and restore processes, the ON-Bar commands, and the ON-Bar configuration parameters are different for IBM Informix Dynamic Server and IBM Informix Extended Parallel Server.

The **ontape** utility does not require a storage manager and works on Dynamic Server only.

Types of Users

This manual is written for the following users:

- Database administrators
- System administrators
- Backup operators
- Technical support personnel

This manual is written with the assumption that you have the following background:

- Some experience with storage managers, which are applications that manage the storage devices and media that contain backups
- A working knowledge of your computer, your operating system, and the utilities that your operating system provides
- Some experience working with relational databases or exposure to database concepts
- Some experience with database server administration, operating-system administration, or network administration

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

Software Dependencies

This manual is written with the assumption that you are using one of the following database servers:

- IBM Informix Dynamic Server, Version 9.4
- IBM Informix Extended Parallel Server, Version 8.40

ON-Bar works differently on Dynamic Server and on Extended Parallel Server.

Assumptions About Your Locale

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

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

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

Demonstration Database

The DB-Access utility, which is provided with the database server products, includes one or more of the following demonstration databases:

- The **stores_demo** database illustrates a relational schema with information about a fictitious wholesale sporting-goods distributor. Many examples in IBM Informix manuals are based on the **stores_demo** database.
- The **sales_demo** database illustrates a dimensional schema for data-warehousing applications. For conceptual information about dimensional data modeling, see the *IBM Informix Database Design and Implementation Guide*. ♦
- The **superstores_demo** database illustrates an object-relational schema. The **superstores_demo** database contains examples of extended data types, type and table inheritance, and user-defined routines. ♦

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

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

XPS

IDS

New Features in Dynamic Server, Version 9.4

The following tables provide information about the new backup and restore features in IBM Informix Dynamic Server, Version 9.4, that this manual covers. For a description of all new features, see the *Getting Started Guide*.

New Features	Reference
Renaming a chunk to a different path and offset during a cold restore	“Renaming Chunks During a Restore” on page 6-27 and page 14-16
Specifying writing to and reading from a tape until the end of the device with ontape	“Specifying the Tape-Size Parameters” on page 12-7
A modifiable shell script, alarmprogram.sh , to handle event alarms	“ALARMPROGRAM” on page 9-7

Features from Dynamic Server 9.3

Version 9.3 includes new features that make the database server easier to install, use, and manage.

New Features	Reference
Dynamic addition of logical logs	“Ensuring That You Have Enough Logical-Log Space” on page 4-9
The ON-Bar option in IBM Informix Server Administrator	ISA online help

Features from Dynamic Server 9.21

These features were introduced in IBM Informix Dynamic Server, Version 9.21.

Features	Reference
Backup and restore of nonlogging (RAW) tables using ON-Bar or ontape	“Backing Up Table Types” on page 4-21 and page 13-12 “Restoring Table Types” on page 6-38 and page 14-16
onbar -b -l command to back up logical logs	“Backing Up Logical Logs” on page 4-21
The log_full.sh and log_full.bat scripts have been updated with the new parameters for continuous log backups. The onbar-l syntax for backing up logical logs will not be supported in the future.	

Organizational Changes to This Manual Since Version 9.2

The *Backup and Restore Guide* has been reorganized as follows:

- [Chapter 1](#) discusses backup and restore concepts and compares ON-Bar with **ontape**.
- [Chapter 2](#) through [Chapter 11](#) cover ON-Bar.
- [Chapter 12](#) through [Chapter 14](#) cover **ontape**.
- [Chapter 7](#) contains new information on using external backup and restore to initialize High-Availability Data Replication.

New Features in Extended Parallel Server

For a list of new features in Extended Parallel Server, Version 8.40 see the *Getting Started Guide* manual.

Documentation Conventions

This section describes the conventions that this manual uses. These conventions make it easier to gather information from this and other volumes in the documentation set.

Typographical Conventions

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

Convention	Meaning
KEYWORD	All primary elements in a programming language statement (keywords) appear in uppercase letters in a serif font.
<i>italics</i> <i>italics</i> <i>italics</i>	Within text, new terms and emphasized words appear in italics. Within syntax and code examples, variable values that you are to specify appear in italics.
boldface <i>boldface</i>	Names of program entities (such as classes, events, and tables), environment variables, file and pathnames, and interface elements (such as icons, menu items, and buttons) appear in boldface.
monospace <i>monospace</i>	Information that the product displays and information that you enter appear in a monospace typeface.

(1 of 2)



Convention	Meaning
KEYSTROKE	Keys that you are to press appear in uppercase letters in a sans serif font.
◆	This symbol indicates the end of product- or platform-specific information.
→	This symbol indicates a menu item. For example, “Choose Tools→Options ” means choose the Options item from the Tools menu.

(2 of 2)

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 three types of information, as the following table describes. This information always appears in italics.

Icon	Label	Description
	Warning:	Identifies paragraphs that contain vital instructions, cautions, or critical information
	Important:	Identifies paragraphs that contain significant information about the feature or operation that is being described
	Tip:	Identifies paragraphs that offer additional details or shortcuts for the functionality that is being described

Feature, Product, and Platform Icons

Feature, product, and platform icons identify paragraphs that contain feature-specific, product-specific, or platform-specific information.

Icon	Description
	Identifies information that relates to the IBM Informix Global Language Support (GLS) feature
	Identifies information or syntax that is specific to IBM Informix Dynamic Server
	Identifies information that is specific to the UNIX operating system
	Identifies information that applies to all Windows environments
	Identifies information or syntax that is specific to IBM Informix Extended Parallel Server

These icons can apply to an entire section or to one or more paragraphs within a section. If an icon appears next to a section heading, the information that applies ends at the next heading at the same or higher level. A ♦ symbol indicates the end of information that appears in one or more paragraphs within a section.

Command-Line Conventions

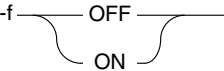
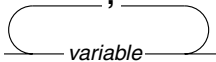
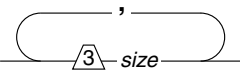
This section defines and illustrates the format of commands that are available in IBM Informix products. These commands have their own conventions, which might include alternative forms of a command, required and optional parts of the command, and so forth.

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 IBM Informix product. It might appear alone or precede one or more options. You must spell a command exactly as shown and use lowercase letters.
<i>variable</i>	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.
<div>Privileges p. 5-17</div> <div>Privileges</div>	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)

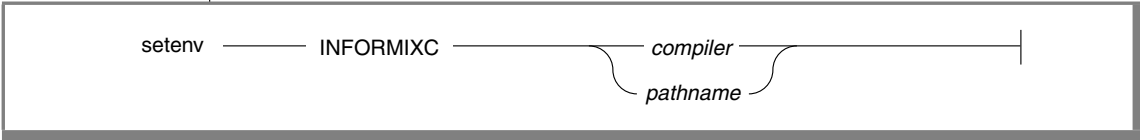
Element	Description
	A branch below the main path indicates an optional path. (Any term on the main path is required, unless a branch can circumvent it.)
	A loop indicates a path that you can repeat. Punctuation along the top of the loop indicates the separator symbol for list items.
	A gate ($\sqrt{3}$) 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. You can specify <i>size</i> 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. Follow the diagram to the right, including the elements that you want. The elements in the diagram are case sensitive.

Figure 1 illustrates the following steps:

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

IBM Informix Dynamic Server documentation is provided in a variety of formats:

- **Online manuals.** The documentation CD in your media pack allows you to print the product documentation. You can obtain the same online manuals at the IBM Informix Online Documentation site at <http://www.ibm.com/software/data/informix/pubs/library/>.
- **Online help.** This facility provides context-sensitive help, an error message reference, language syntax, and more.
- **Documentation notes and release notes.** Documentation notes, which contain additions and corrections to the manuals, and release notes are located in the directory where the product is installed.

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

UNIX

On UNIX platforms, the following online files appear in the \$INFORMIXDIR/release/en_us/0333 directory.

Online File	Purpose
ids_onbar_docnotes_9.40.html xps_onbar_docnotes_8.40.html ids_onbar_docnotes_9.40.txt xps_onbar_docnotes_8.40.txt	The documentation notes file for your version of this manual describes topics that are not covered in the manual or that were modified since publication.
ids_unix_release_notes_9.40.html xps_release_notes_8.40.html ids_unix_release_notes_9.40.txt xps_release_notes_8.40.txt	The release notes file describes feature differences from earlier versions of IBM Informix products and how these differences might affect current products. This file also contains information about any known problems and their workarounds.
ids_machine_notes_9.40.txt xps_machine_notes_8.40.txt	The machine notes file describes any special actions that you must take to configure and use IBM Informix products on your computer. Machine notes are named for the product described.



Windows

The following items appear in the **Informix** folder. To display this folder, choose **Start→Programs→Informix→ Documentation Notes** or **Release Notes** from the task bar.

Program Group Item	Description
Documentation Notes	This item includes additions or corrections to manuals with information about features that might not be covered in the manuals or that have been modified since publication.
Release Notes	This item describes feature differences from earlier versions of IBM Informix products and how these differences might affect current products. This file also contains information about any known problems and their workarounds.

Machine notes do not apply to Windows platforms. ♦

- **Error message files.** IBM Informix software products provide ASCII files that contain error messages and their corrective actions.

To read the error messages on UNIX, use the **finderr** command to display error messages online. ♦

To read error messages and corrective actions on Windows, use the **Informix Error Messages** utility. To display this utility, choose **Start→Programs→Informix** from the task bar. ♦

UNIX

Windows

Related Reading

For a list of publications that provide an introduction to database servers and operating-system platforms, refer to your *Getting Started Guide*.

Compliance with Industry Standards

The American National Standards Institute (ANSI) has established a set of industry standards for SQL. IBM Informix SQL-based products are fully compliant with SQL-92 Entry Level (published as ANSI X3.135-1992), which is identical to ISO 9075:1992. In addition, many features of Informix database servers comply with the SQL-92 Intermediate and Full Level and X/Open SQL CAE (common applications environment) standards.

IBM Welcomes Your Comments

To help us with future versions of our manuals, let us know about any corrections or clarifications that you would find useful. Include the following information:

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

Send electronic mail to us at the following address:

`docinf@us.ibm.com`

This address is reserved for reporting errors and omissions in our documentation. For immediate help with a technical problem, contact Customer Services.

Introducing Backup and Restore

Chapter 1 **Introducing Backup and Restore**

Section I



Introducing Backup and Restore

In This Chapter	1-3
What Is a Recovery System?.	1-3
What Is a Backup?.	1-4
What Is a Logical-Log Backup?	1-5
When You Do Not Use Logging.	1-6
What Are Manual and Continuous Logical-Log Backups?.	1-6
What Is a Log Salvage?.	1-6
Why You Need to Save Logical-Log Backups	1-6
What Is a Restore?.	1-8
What Are Warm, Cold, and Mixed Restores?	1-8
What Are Physical and Logical Restores?	1-10
Comparing ON-Bar and ontape	1-12
Planning a Recovery Strategy	1-14
What Types of Data Loss Can Occur?	1-14
How Severe is the Failure?	1-15
How Do You Use Your Data?	1-16
Scheduling Backups	1-17
Planning a Backup System for a Production Database Server	1-18
Evaluating Hardware and Memory Resources	1-18
Evaluating Backup and Restore Time	1-18
Evaluating Logging and Transaction Activity	1-20

In This Chapter

Two utilities are provided for backing up and restoring database server data.

Utility	Storage Manager	Where Discussed
ON-Bar	IBM Informix Storage Manager (ISM)	Chapter 2 through Chapter 11
ontape	None	Chapter 12 through Chapter 14

This chapter explains basic backup and restore concepts for Informix database servers and covers the following topics:

- Comparing ON-Bar and **ontape**
- Planning a recovery strategy
- Planning a backup system for a production database server

What Is a Recovery System?

A *recovery system* enables you to back up your database server data and subsequently restore it if your current data becomes corrupt or inaccessible. The causes of data corruption or loss can range from a program error to a disk failure 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 a Backup?

A *backup* is a copy of one or more *dbspaces* (also called storage spaces) and logical logs that the database server maintains. On Dynamic Server, you can also back up *blobspaces* and *sbspaces*. On Extended Parallel Server, you can also back up *dbsllices*. For a description of storage spaces, see your *Administrator's Guide*.

The backup copy is usually written to a *secondary storage* medium such as disk, magnetic tape, or optical disk. We recommend that you store the media offline and keep a copy off-site if possible.

Important: Database backups do not replace ordinary operating-system backups, which back up files other than Informix database files.

Figure 1-1 illustrates the basic concept of a database backup.

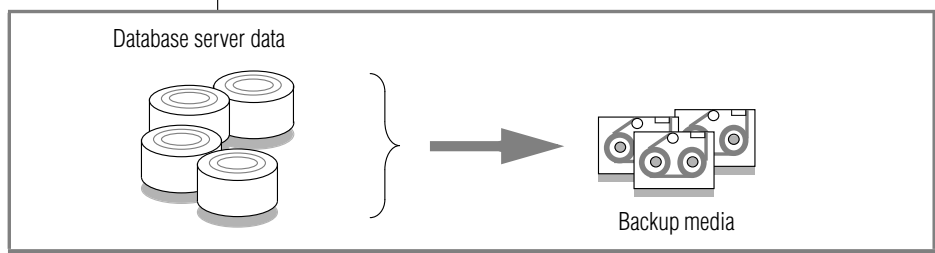


Figure 1-1
*A Backup of
Database Server
Data*

You do not always have to back up all the storage spaces. If some tables change daily but others rarely change, it is inefficient to back up the storage spaces that contain the unchanged tables every time that you back up the database server. You need to plan your backup schedule carefully to avoid long delays for backing up or restoring data.

To provide a more flexible backup environment, ON-Bar and **ontape** support the following three *backup levels*:

- Level 0 backs up all used pages that contain data for the specified storage spaces.

You need all these pages to restore the database to the state that it was in at the time that you made the backup.

- Level 1 backs up only data that has changed since the last level-0 backup of the specified storage spaces.

All changed table and index pages are backed up, including those with deleted data. The data that is copied to the backup reflects the state of the changed data at the time that the level-1 backup began.

- Level 2 backs up only data that has changed since the last level-1 backup of the specified storage spaces.

A level-2 backup contains a copy of every table and index page in a storage space that has changed since the last level-1 backup.



Important: *If disks and other media are completely destroyed and need to be replaced, you need at least a level-0 backup of all storage spaces and relevant logical logs to restore data completely on the replacement hardware.*

For details, see [Chapter 4, “Backing Up with ON-Bar,”](#) and [“Backing Up with ontape”](#) on page 13-1.

What Is a Logical-Log Backup?

A *logical-log backup* is a copy to disk or tape of all full logical-log files. The logical-log files store a record of database server activity that occurs *between backups*.

To free full logical-log files, back them up. The database server reuses the freed logical-log files for recording new transactions. For a complete description of the logical log, see your *Administrator’s Guide*.

When You Do Not Use Logging

Even if you do not specify logging for databases or tables, you need to back up the logical logs because they contain 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.

What Are Manual and Continuous Logical-Log Backups?

A *manual logical-log backup* backs up all the full logical-log files and stops at the current logical-log file.

If you turn on *continuous logical-log backup*, the database server backs up each logical log automatically when it becomes full. If you turn off continuous logical-log backup, the logical-log files continue to fill. If all logical logs are filled, the database server hangs until the logs are backed up.

What Is a Log Salvage?

When the database server is offline (Dynamic Server) or in microkernel mode (Extended Parallel Server), you can perform a special kind of logical-log backup, called a *log salvage*. In a log salvage, the database server accesses the log files directly from disk. The log salvage backs up any logical logs that have not yet been backed up and are not corrupted or destroyed. The log salvage enables you to recover all of your data up to the last available and uncorrupted logical-log file and the last complete transaction.

Why You Need to Save Logical-Log Backups

Perform frequent logical-log backups for the following reasons:

- To free full logical-log files
- To minimize data loss if a disk that contains logical logs fails
- To ensure that restores contain consistent and the latest transactions



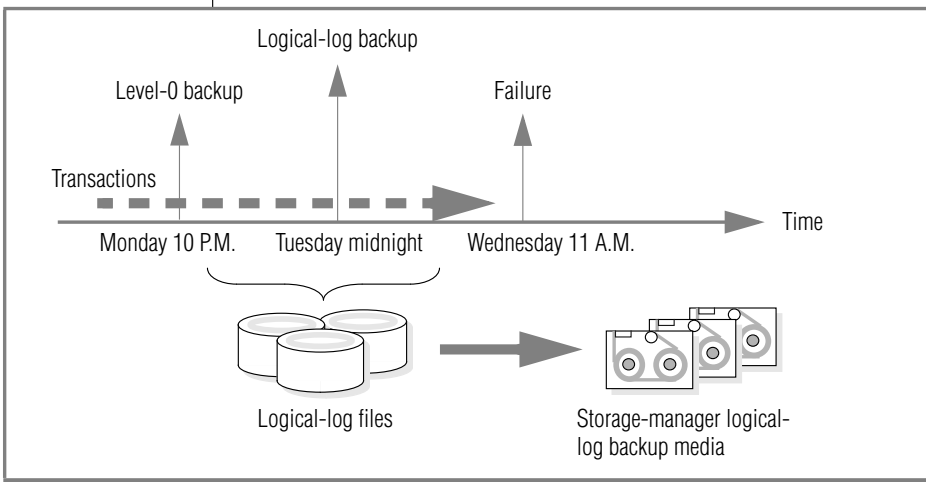
Save the logical-log backups from the last two level-0 backups so that you can use them to complete a restore. If a level-0 backup is inaccessible or unusable, you can restore data from an older backup, if you have one. If any of the logical-log backups are also inaccessible or unusable, however, you cannot roll forward the transactions from those logical-log files or from any subsequent logical-log files.

Warning: You will lose transactions in logical-log files that are not backed up or salvaged.

To illustrate, as [Figure 1-2](#) shows, suppose you perform a level-0 backup on Monday at 10:00 P.M. and then back up the logical logs on Tuesday at midnight. On Wednesday at 11:00 A.M., you suffer a mishap that destroys your databases. You would be unable to restore the transactions that occurred between midnight on Tuesday and 11:00 A.M. on Wednesday unless you had continuous logical-log backup set up.

If the disks that contain the storage spaces with the logical logs are damaged, the transactions after midnight on Tuesday might be lost. To restore these transactions from the last logical-log backup, try to salvage the logical logs before you repair or replace the bad disk and then perform a cold restore.

Figure 1-2
Storage Space and
Logical-Log
Backups



For more information, see [“Backing Up Logical Logs” on page 4-21](#) and [“Backing Up Logical-Log Files with ontape” on page 13-14](#).

What Is a Restore?

A *restore* re-creates database server data from backed-up storage spaces and logical-log files. A restore re-creates database server data that has become inaccessible because of any of the following conditions:

- You need to replace a failed disk that contains database server data.
- A logic error in a program has corrupted a database.
- You need to move your database server data to a new computer.
- A user accidentally corrupted or destroyed data.

To restore data up to the time of the failure, you must have at least one level-0 backup of each of your storage spaces from before the failure and the logical-log files that contain all transactions since these backups.

What Are Warm, Cold, and Mixed Restores?

When you restore data, you must decide whether to do so while the database server is in quiescent, online, offline (Dynamic Server), or microkernel mode (Extended Parallel Server). The types of restores are as follows:

- If you restore noncritical data while the database server is online or quiescent, that process is called a *warm restore*.
- When Dynamic Server is offline or Extended Parallel Server is in microkernel mode, you can perform a *cold restore*.
- A *mixed restore* is a cold restore of some storage spaces followed by a warm restore of the remaining storage spaces.

Warm Restore

As [Figure 1-3](#) shows, a warm restore restores noncritical storage spaces. A warm restore consists of one or more physical restores, a logical-log backup, and a logical restore.

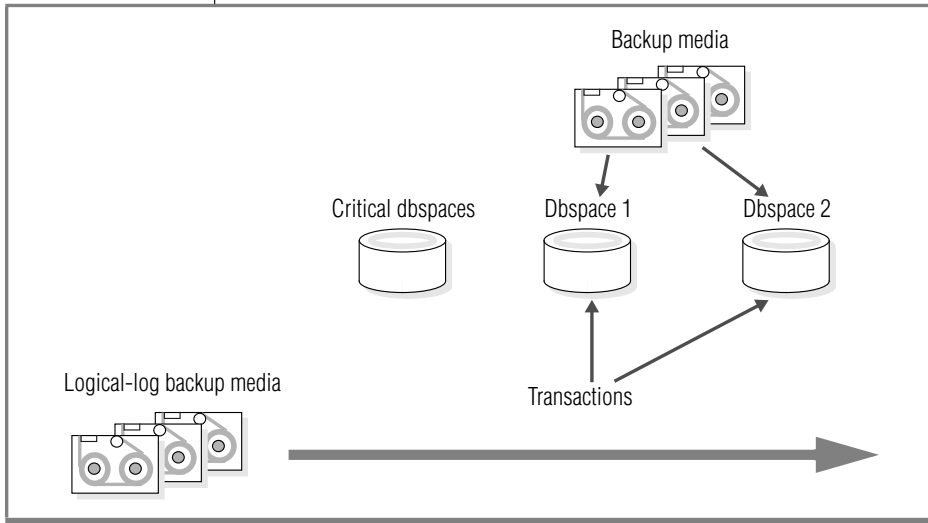


Figure 1-3
Warm Restore

Cold Restore

As [Figure 1-4](#) shows, a cold restore salvages the logical logs, and restores the critical dbspaces (root dspace and the dbspaces that contain the physical log and logical-log files), other storage spaces, and the logical logs.

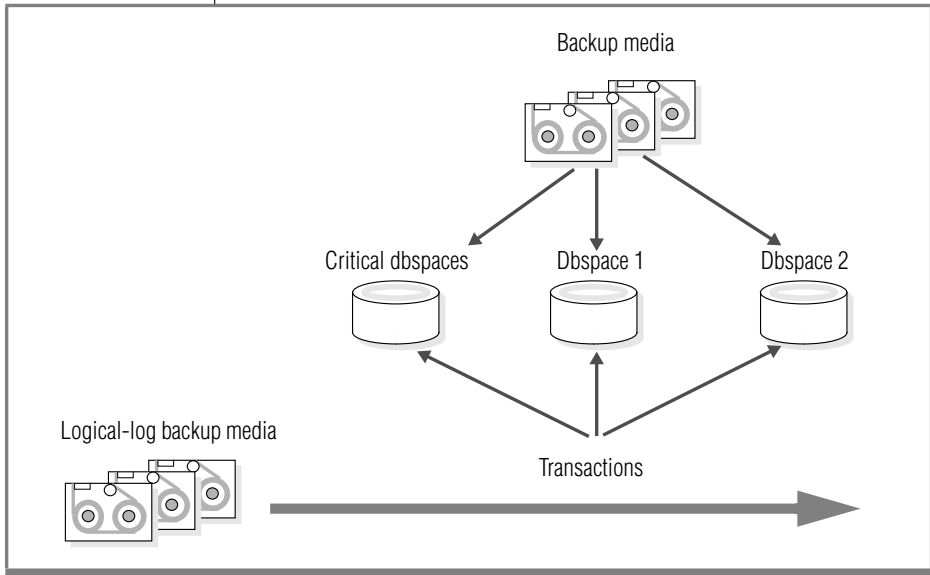


Figure 1-4
Cold Restore

You can perform a cold restore onto a computer that is not identical to the one on which the backup was performed by giving any chunk a new pathname and offset during the restore.

What Are Physical and Logical Restores?

ON-Bar and **ontape** restore database server data in two phases:

- The first phase is the *physical restore*, which restores data from backups of all or selected storage spaces.
- The second phase is the *logical restore*, which restores transactions from the logical-log backups. The database server automatically knows which logical logs to restore.

Physical Restore

During a physical restore, ON-Bar or **ontape** restores the data from the most recent level-0, level-1, and level-2 backups. When you suffer a disk failure, you can restore to a new disk only those storage spaces with chunks that resided on the failed disk. [Figure 1-5](#) illustrates a physical restore.

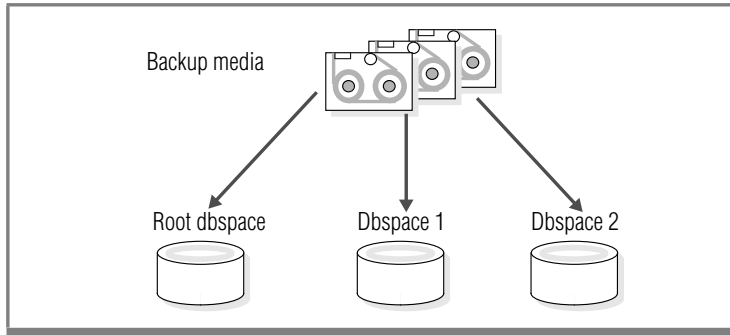


Figure 1-5
Physical Restore

Logical Restore

As [Figure 1-6](#) shows, the database server *replays* the logical logs to reapply any database transactions that occurred after the last backup. The logical restore applies only to the physically-restored storage spaces.

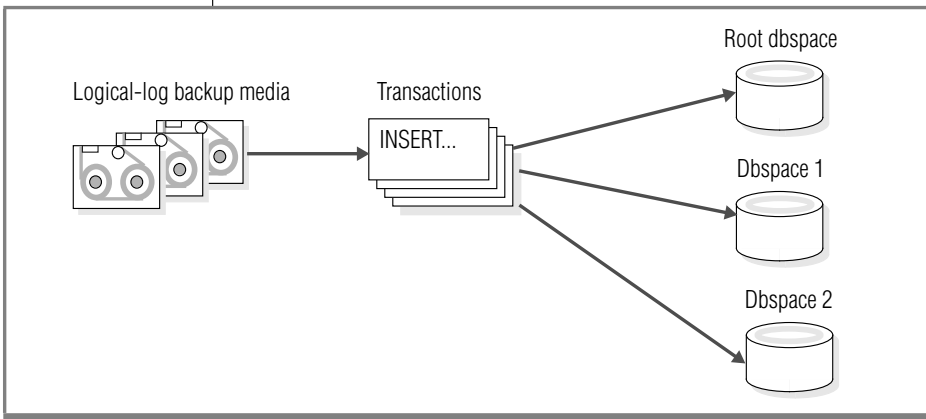


Figure 1-6
Logical Restore

For more information, see [Chapter 6, “Restoring Data with ON-Bar”](#) and [Chapter 14, “Restoring with ontape.”](#)

Comparing ON-Bar and ontape

Figure 1-7 compares ON-Bar and **ontape**. If you are switching to ON-Bar and ISM from **ontape**, note that ON-Bar works differently.

Figure 1-7
Differences Between ON-Bar and ontape

Can the utility...	ON-Bar	ontape
Use a storage manager to track backups and storage media?	yes	no
Back up all database server data?	yes	yes
Back up selected storage spaces?	yes	no
Back up logical-log files?	yes	yes
Perform continuous logical-log backups?	yes	yes
Back up while the database server is online?	yes	yes
Back up while the database server is in quiescent mode?	yes	yes
Restore all database server data?	yes	yes
Restore selected storage spaces?	yes	yes
Back up and restore storage spaces serially?	yes	yes
Perform cold restores with the database server offline or in micro-kernel mode?	yes	yes
Initialize high-availability data replication?	yes	yes
Restore data to a specific point in time?	yes	no
Perform separate physical and logical restores?	yes	no
Back up and restore different storage spaces in parallel?	yes	no
Use multiple tape drives concurrently for backups and restores?	yes	no
Restart a restore?	yes	no
Rename a chunk pathname or device during a cold restore?	yes	yes

(1 of 2)

Can the utility...	ON-Bar	ontape
Perform imported restores?	yes	no
Perform external backups and restores?	yes	no
Change logging mode for databases?	no	yes

(2 of 2)

The **ontape** utility does not use the **sysutils** database or the emergency boot files. The **ontape** utility supports remote backup devices on other hosts but ISM does not. ISM supports different sets of tape drives on various hardware platforms.

The **ontape** utility supports two simultaneous sessions, one for physical backup or restore, and one for log backup. Each ISM instance has a limit of four simultaneous sessions.

You can use ON-Bar with third-party storage managers to obtain more sophisticated device support and storage management.

The **ontape** utility allows you to change the logging mode of a database. If you use ON-Bar, use **ondblog** to change the logging mode of a database.

Warning: *The backup tapes that **ontape** and ON-Bar produce are not compatible! You cannot create a backup with **ontape** and restore it with ON-Bar, or vice versa.*



Planning a Recovery Strategy

Before you use ON-Bar or **ontape**, plan your recovery goals.

What Types of Data Loss Can Occur?

The first step is to determine how much data loss, if any, is acceptable. The following types of data loss can occur:

- Deletion of the following:
 - Rows, columns, tables, or databases
 - Chunks, storage spaces, or logical logs
- Data corruption or incorrect data created
- Hardware failure (such as a disk that contains chunk files fails or a backup tape that wears out)
- Database server failure
- Natural disaster

How Severe is the Failure?

After you determine your recovery goals, create your recovery plan. Develop a recovery plan for multiple levels of failure, as [Figure 1-8](#) shows.

Figure 1-8
Sample Recovery Plans

Failure Severity	Data Loss	Suggested Recovery Plan
Small	Noncritical data is lost.	Restore of the data can wait until a nonpeak time. Use a warm restore.
Medium	The data that is lost is critical for your business but does not reside in a critical dbspace.	Perform a warm restore of this data as soon as possible.
Large	Critical dbspaces are lost.	Use a mixed restore to restore the critical data right away and a warm restore to restore noncritical data during off-peak hours.
Disaster	All data is lost.	Perform a cold or mixed restore as soon as possible.

How Do You Use Your Data?

After you develop your recovery plan, create a backup plan. How you use the data also determines how you plan your backup schedule, as follows:

- **Data usage**

How do users use the data?

- Critical dbspaces (root dbspace and dbspaces that contain the physical log and at least one logical-log file)
- Critical business application data
- Long-term data storage for legal or record-keeping reasons
- Data sharing among groups
- Test data

- **Transaction Time**

How much transaction time can be lost? Also, how long might it take to re-enter lost transactions manually? For example, can you afford to re-enter all transactions that occurred over the past three hours?

- **Quantity and Distribution**

How much data can you afford to lose? For example, you lost one fourth of your customer profiles, or you lost the Midwest regional sales figures but the West Coast figures are intact.

Ask the following questions to assist in deciding how often and when you want to back up the data:

- Does your business have down time where the system can be restored?
- If your system is 24x7 (no down time), is there a nonpeak time where a restore could occur?
- If a restore must occur during a peak period, how critical is the time?
- Which data can you restore with the database server online (warm restore)? Which data must be restored offline (cold restore)?
- How many storage devices are available to back up and restore the data?

Scheduling Backups

Figure 1-9 shows a sample backup plan for a small or medium-sized system. Tailor your backup plan to the requirements of your system. The more often the data changes and the more important it is, the more frequently you need to back it up. For more information, see “Choosing a Backup Level” on page 4-8.

Figure 1-9
Sample Backup Plan

Backup Level	Backup Schedule
Complete (level-0) backup	Saturday at 6 P.M.
Incremental (level-1) backup	Tuesday and Thursday at 6 P.M.
Incremental (level-2) backup	Daily at 6 P.M.
Level-0 backup of storage spaces that are updated frequently	Hourly



Important: Perform a level-0 backup after you change the physical schema, such as adding a chunk to a storage space. (See “Collecting Information About Your System Before a Backup” on page 4-9.)

Planning a Backup System for a Production Database Server

To plan for adequate backup protection for your data, analyze your database server configuration and activity and the types of backup media available at your installation. Also, consider your budget for storage media, disks, computers and controllers, and the size of your network.

Evaluating Hardware and Memory Resources

Evaluate the following database server and hardware configuration elements to determine which storage manager and storage devices to use:

- The number of I/O virtual processors
- The amount of memory available and the distribution of processor activity

Evaluating Backup and Restore Time

How long your backup or restore takes depends on your database server configuration and the database size:

- The speed of disks or tape devices
The faster the storage devices, the faster the backup or restore time.
- The number of incremental backups that you want to restore if a disk or system failure requires you to rebuild the database
Incremental backups use less storage space than full backups and also reduce restore time.
- The size and number of storage spaces in the database
Backups: Many small storage spaces take slightly longer to back up than a few large storage spaces of the same total size.
Restores: A restore usually takes as long to recover the largest storage space and the logical logs.

- Whether storage spaces are mirrored

If storage spaces are mirrored, you reduce the chance of having to restore damaged or corrupted data. You can restore the mirror at nonpeak time with the database server online.
- The length of time users are interrupted during backups and restores

If you perform backups and warm restores while the database server is online, users can continue their work but might notice a slower response. If you perform backups and warm restores with the database server in quiescent mode, users must exit the database server. If you perform a cold restore with the database server offline, the database server is unavailable to users, so the faster the restore, the better. An external backup and restore eliminates system downtime.
- The backup schedule

Not all storage spaces need to be included in each backup or restore session. Schedule backups so that you can back up more often the storage spaces that change rapidly than those that seldom or never change. Be sure to back up each storage space at level-0 at least once.
- The layout of the tables across the dbspaces and the layout of dbspaces across the disks

When you design your database server schema, organize the data so that you can restore important information quickly. For example, you should isolate critical and frequently used data in a small set of storage spaces on the fastest disks. You also can fragment large tables across dbspaces to balance I/O and maximize throughput across multiple disks. For more information, see your *Performance Guide*.
- The database server and system workload

The greater the workload on the database server or system, the longer the backup or restore time.
- The values of backup and restore configuration parameters

For example, the number and size of data buffers that ON-Bar uses to exchange data with the database server can affect performance. Use the `BAR_NB_XPORT_COUNT` and `BAR_XFER_BUF_SIZE` (IDS) or `BAR_XPORT_COUNT` and `BAR_XFER_BUFSIZE` (XPS) configuration parameters to control the number and size of data buffers.

Evaluating Logging and Transaction Activity

The following database server usage requirements also affect your decisions about the storage manager and storage devices:

- The amount and rate of transaction activity that you expect
- The number and size of logical logs
If you need to restore data from a database server with very little transaction activity, define many small logical logs. You are less likely to lose data because of infrequent logical-log backups.
- How fast the logical-log files fill
Back up log files before they fill so that the database server does not hang
- Database and table logging modes
When you use many nonlogging databases or tables, logical-log backups might become less frequent.

The ON-Bar Backup and Restore System

Chapter 2	The ON-Bar Backup and Restore System
Chapter 3	Configuring the Storage Manager and ON-Bar
Chapter 4	Backing Up with ON-Bar
Chapter 5	Verifying Backups
Chapter 6	Restoring Data with ON-Bar
Chapter 7	Performing an External Backup and Restore
Chapter 8	Using ONBar Utilities
Chapter 9	Setting ON-Bar Configuration Parameters
Chapter 10	ON-Bar Catalog Tables
Chapter 11	ON-Bar Messages and Return Codes



The ON-Bar Backup and Restore System

In This Chapter	2-3
Where to Find Information on ON-Bar and ISM	2-4
ON-Bar for Dynamic Server	2-6
ON-Bar for Extended Parallel Server	2-9
Database Server and Storage-Manager Communication	2-9
Backup Scheduler	2-10
ON-Bar Utilities	2-12
IBM Informix Storage Manager	2-13
Third-Party Storage Managers	2-13
The XBSA Interface	2-14
The ON-Bar Tables	2-15
The Emergency Boot Files	2-15
Emergency Boot File on Dynamic Server	2-15
Emergency Boot Files on Extended Parallel Server	2-16
The ON-Bar Activity Log	2-17
Specifying the Location of the Activity Log	2-17
Monitoring the Progress of a Backup or Restore	2-17

In This Chapter

This chapter introduces the components of ON-Bar and describes how it works. The following topics are covered:

- Where to find information on ON-Bar and ISM
- ON-Bar for Dynamic Server
- ON-Bar for Extended Parallel Server
- ON-Bar utilities

Figure 2-1 shows which database server versions support ON-Bar and IBM Informix Storage Manager (ISM), Version 2.2.

Figure 2-1
On-Bar and ISM Support

Database Server	Version	ON-Bar Support	ISM Support
Dynamic Server	Version 7.24	X	
Dynamic Server	Version 7.3x	X	X
IBM Informix Universal Server	Version 9.1x	X	X
Dynamic Server	Version 9.2x	X	X
Dynamic Server	Version 9.3	X	X
Dynamic Server	Version 9.4	X	X
IBM Informix OnLine XPS	Version 8.11	X	X

(1 of 2)

Database Server	Version	ON-Bar Support	ISM Support
Dynamic Server with AD and XP Options	Version 8.2x	X	X
Extended Parallel Server	Version 8.3x	X	X
Extended Parallel Server	Version 8.40	X	X

(2 of 2)

Where to Find Information on ON-Bar and ISM

The task-documentation matrix in [Figure 2-2](#) provides a quick reference to locating ON-Bar commands and ISM information.

Figure 2-2
ON-Bar and ISM Task-Documentation Matrix

If You Want To:	Chapter or Manual
Learn backup and restore concepts	Chapter 1
Configure and use ON-Bar and ISM or another storage manager	Chapter 3 of this manual <i>IBM Informix Storage Manager Administrator's Guide</i> Third-party storage-manager manual
Use the onbar script to customize ON-Bar and ISM operations	Chapter 3 (setup) Chapter 8 (customization)
Use ON-Bar and ISM configuration parameters See a list of the files that ON-Bar and ISM use	Chapter 9 of this manual <i>IBM Informix Storage Manager Administrator's Guide</i>
Set up ISM to use certain storage devices for backup and restore operations Manage backup media and storage devices for ON-Bar Track the location of all backup data Move backup data through a managed life cycle	<i>IBM Informix Storage Manager Administrator's Guide</i> or third-party storage-manager manual

(1 of 3)

If You Want To:	Chapter or Manual
Back up storage spaces and logical logs: <ul style="list-style-type: none">■ onbar -b -L [0 1 2] (standard backup)■ onbar -b -O (override error checking)■ onbar -b -w (whole-system backup, IDS)■ onbar -b -F (fake backup, IDS)■ onbar -b -p (physical backup, XPS)	“Backing Up Storage Spaces and Logical Logs” on page 4-10
Back up logical logs only: <ul style="list-style-type: none">■ onbar -b -l■ onbar -b -l -s (log salvage)■ onbar -b -l -c (backup includes current log, IDS)■ onbar -b -l -C (continuous log backup, IDS)	“Backing Up Logical Logs” on page 4-21
Verify backups before you use the data in a restore: <ul style="list-style-type: none">■ onbar -v (verify backup)■ onbar -b -v (backup and verify, XPS)	Chapter 5
Perform warm or cold restores: <ul style="list-style-type: none">■ onbar -r (parallel restore)■ onbar -r -p (physical restore)■ onbar -r -l (logical restore)■ onbar -r -O (override error checking)■ onbar -r -t (point-in-time restore)■ onbar -r -n (point-in-log restore, IDS)■ onbar -r -w (whole-system restore, IDS)■ onbar -RESTART (restartable restore, IDS)■ onbar -r rename (rename chunks restore, IDS)	Chapter 6
Perform external backups and restores: <ul style="list-style-type: none">■ onmode -c block unblock (external backup, IDS)■ onutil ebr block unblock (external backup, XPS)■ onbar -r -e (external restore)	Chapter 7
Use the onsmsync utility to expire old backup objects	Chapter 8

If You Want To:	Chapter or Manual
Use the onbar_w utility or start_worker script to start onbar-worker processes manually (XPS)	Chapter 8
Monitor Backup Scheduler status (XPS)	Chapter 8
Refer to the tables in the sysutils database and the Backup Scheduler tables in the sysmaster database	Chapter 10
Find corrective actions to ON-Bar error messages	<i>IBM Informix Error Messages</i> at http://www-3.ibm.com/software/data/informix/pubs/library/ or finderr
Find ON-Bar return codes	Chapter 11
Use GLS with ON-Bar	Appendix D
Create and delete storage spaces and chunks	<i>Administrator's Guide</i> for your database server
Manage database-logging status, logical-log files, and the physical log	
Perform fast recovery	
Locate complete information on all database server configuration parameters	<i>Administrator's Reference</i>
Use the ondblog utility to change the logging mode	
Use the onlog utility to display logical-log records	

(3 of 3)

ON-Bar for Dynamic Server

Figure 2-3 on page 2-8 shows the following components of ON-Bar for Dynamic Server:

- *Storage spaces* (dbspaces, blobspaces, and sbspaces) and logical logs to be backed up or restored
- The ON-Bar catalog tables in the **sysutils** database
- The **onbar** script (**onbar.sh** on UNIX or **onbar.bat** on Windows)
- The **onbar-driver** (**onbar_d**)

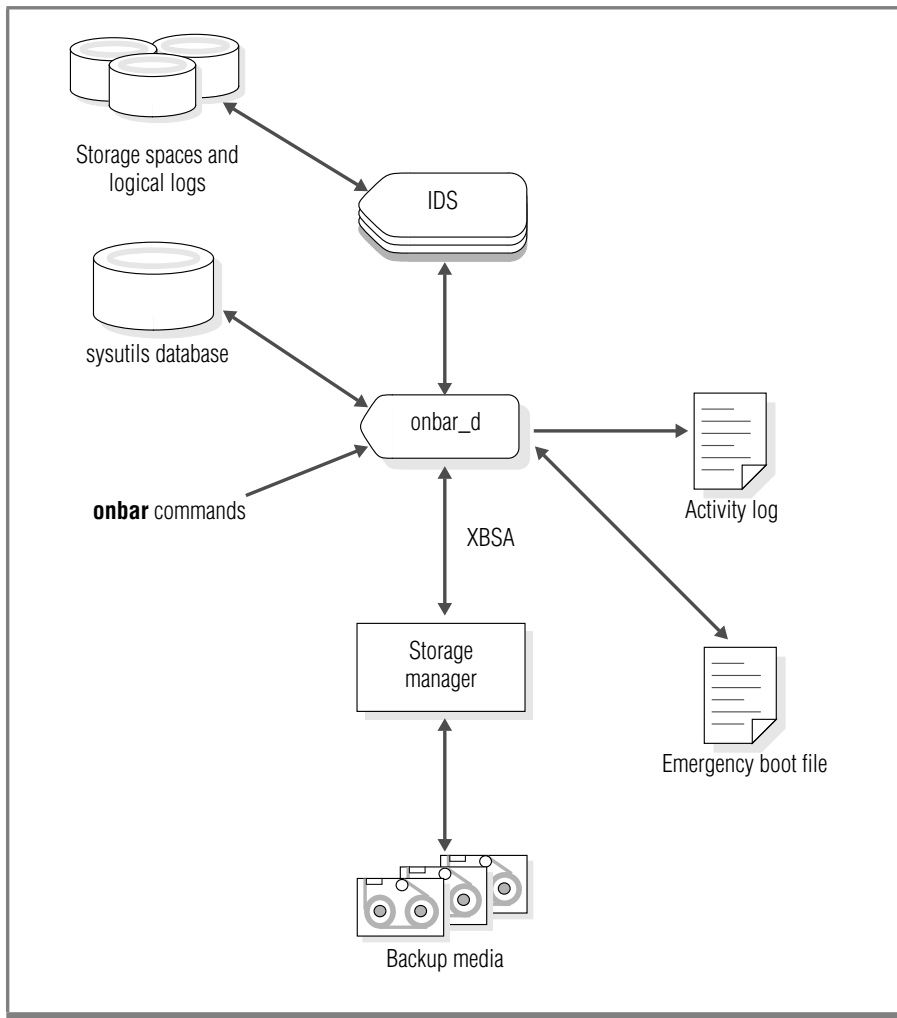
- The XBSA shared library for the storage manager on your system
Use either ISM or a storage manager that a third-party vendor provides.
- Backup data on storage media
- The ON-Bar activity log
- The ON-Bar emergency boot file

ON-Bar communicates with both the database server and the storage manager. Use the **onbar** command to start a backup or restore. For a backup session, ON-Bar requests the contents of storage spaces and logical logs from the database server and passes them to the storage manager. The storage manager stores the data on storage media. For a restore session, ON-Bar requests the backed up data from the storage manager and restores it on the database server.

If you specify a parallel backup or restore, the **onbar-driver (onbar_d)** creates child **onbar_d** processes that perform backup and restore operations. Each child processes one storage space, then returns. ON-Bar processes log files serially. If you specify a serial backup or restore, the **onbar-driver** performs the operation one object at a time.

The **onbar_d** processes write status and error messages to the ON-Bar activity log and write information to the emergency boot file that is used in a cold restore. For more details, see [“Backup Sequence on Dynamic Server” on page 4-30](#).

Figure 2-3
ON-Bar Components for Dynamic Server



ON-Bar for Extended Parallel Server

Figure 2-4 on page 2-11 shows the following components of ON-Bar for Extended Parallel Server:

- Storage spaces (dbspaces and dbslices) and logical logs to be backed up or restored
- The ON-Bar catalog tables in the **sysutils** database
- The **onbar** script and **onbar-driver** (**onbar_d**), **onbar-worker** (**onbar_w**), and **onbar-merger** (**onbar_m**)
- The XBSA shared library for each storage manager on your system
Use either ISM or a storage manager that a third-party vendor provides.
- Backup data on storage media
- The ON-Bar activity log
- The ON-Bar emergency boot files

Database Server and Storage-Manager Communication

For a backup session, ON-Bar requests the contents of storage spaces and logical logs from the database server and passes them to the storage manager. The storage manager stores the data on storage media. For a restore session, ON-Bar requests the backed-up data from the storage manager and restores it on the database server.

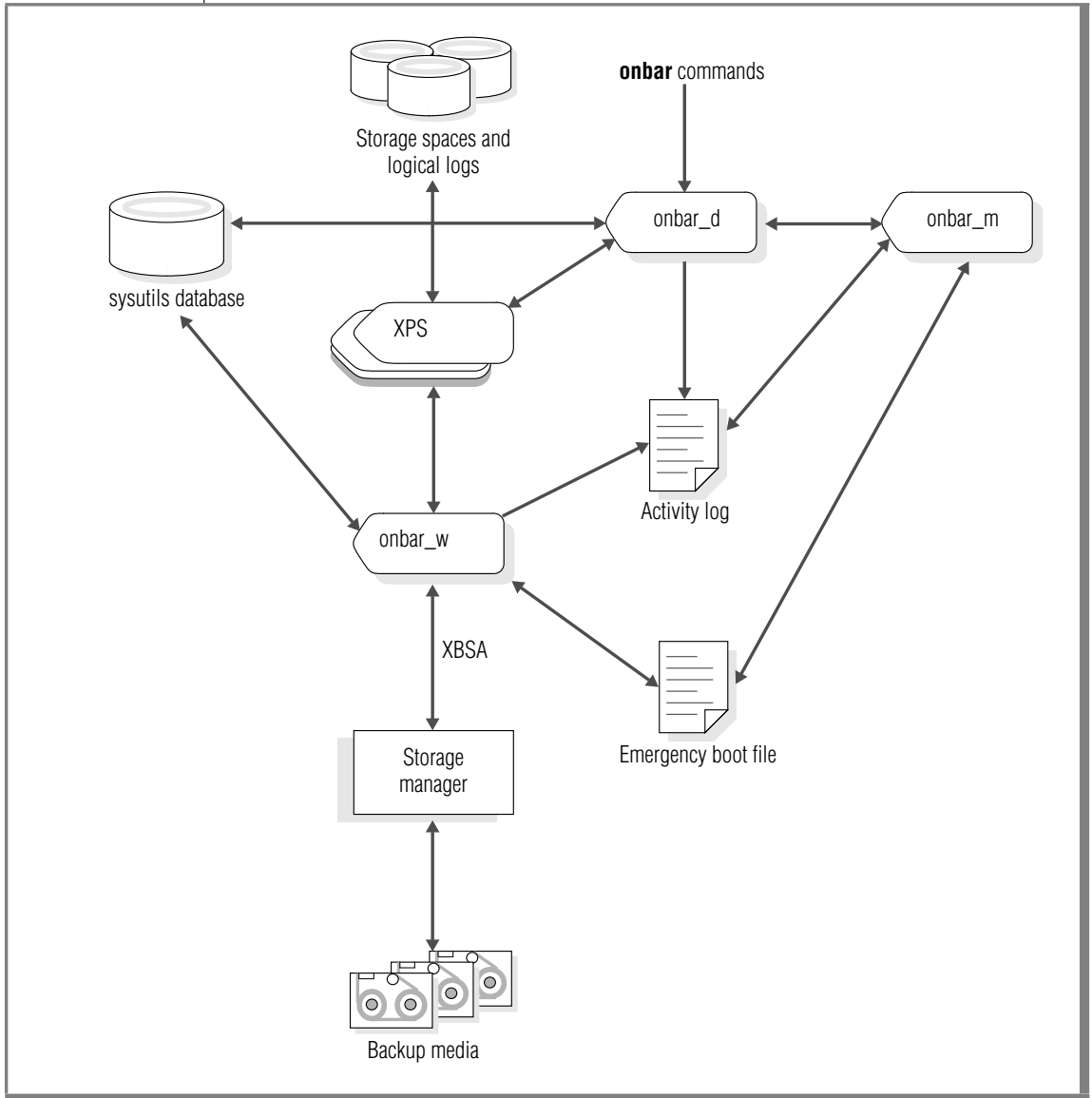
The **onbar_d**, **onbar_w**, and **onbar_m** processes write status and error messages to the ON-Bar activity log. The **onbar_w** and **onbar_m** processes write information to the emergency boot files that are used in a cold restore.

Backup Scheduler

The **onbar-driver** (**onbar_d**) communicates backup or restore requests to the Backup Scheduler on Extended Parallel Server. The *Backup Scheduler* tracks all active and scheduled backup and restore activities for all the coservers. The Backup Scheduler creates one or more sessions, each of which contains a list of objects to back up or restore. It starts **onbar-worker** processes to back up or restore the objects and coordinates the session activity. A *session* is a single backup or restore request.

For details, see [“Backup Sequence on Extended Parallel Server”](#) on page 4-32.

Figure 2-4
ON-Bar Components for Extended Parallel Server



ON-Bar Utilities

ON-Bar includes the following utilities. You can call **onbar** from the command line, a script, a scheduler such as **cron** (UNIX), or a storage-manager process.

Utility	Description	IDS	XPS
onbar	Is an editable shell script on UNIX and a batch file (onbar.bat) in Windows that starts the onbar-driver . Use the onbar script or batch file to check the storage-manager version and customize backup and restore operations.	✓	✓
onbar_d	When you use the onbar command, it calls the onbar_d utility that starts the onbar-driver . The onbar-driver starts and controls backup and restore activities. The onbar_d utility transfers data between Dynamic Server and the storage manager.	✓	✓
onbar_w	Transfers data between an Extended Parallel Server coserver and the storage manager until the backup or restore request is fulfilled.		✓
onbar_m	Collects and processes the backup emergency boot files from each coserver and creates the restore boot file.		✓
start_worker.sh	Starts onbar-worker processes manually.		✓
onsmsync	Cleans up old backup history in the sysutils database and emergency boot files.	✓	✓
ondblog	Changes the database-logging mode.	✓	✓

IBM Informix Storage Manager

ON-Bar is packaged with ISM. However, you can purchase a third-party storage manager if you prefer. You must use a storage manager to perform backups and restores with ON-Bar. The *storage manager* is an application that manages the storage devices and media that contain backups. The storage manager handles all media labeling, mount requests, and storage volumes.

The ISM server resides on the same computer as ON-Bar and the Informix database server; your storage devices are attached to this computer as well. ISM can store data on simple tape drives, optical disk devices, and file systems. ISM also performs the following functions:

- Configures up to four storage devices
- Adds, changes, and deletes administrative users
- Labels and mounts storage volumes on your storage devices
- Manages storage volumes
- Compresses and decompresses data
- Encrypts and decrypts data

For more information, see [“Installing and Configuring a Third-Party Storage Manager” on page 3-3](#), [“Choosing Storage Managers and Storage Devices” on page 3-20](#), [Chapter 9](#), [“Setting ON-Bar Configuration Parameters,”](#) and the *IBM Informix Storage Manager Administrator’s Guide*.

Third-Party Storage Managers

Some third-party storage managers can manage stackers, robots, and jukeboxes as well as simple tape and disk devices. These storage managers might perform these additional functions:

- Schedule backups
- Support networked and distributed backups and restores

Find information on the third-party storage managers that ON-Bar supports at <http://www.ibm.com/software/data/informix/pubs/smv/index.html>.

Make sure that the storage manager has passed the Informix validation process. The validation process is specific to the backup and restore product version, the operating-system version, and the Informix database server version.

The XBSA Interface

ON-Bar and the storage manager communicate through the X/Open Backup Services Application Programmer's Interface (XBSA), which enables the storage manager to manage media for the database server. By using an open-system interface to the storage manager, ON-Bar can work with a variety of storage managers that also use XBSA.

Each storage manager develops and distributes a unique version of the XBSA shared library. You must use the version of the XBSA shared library provided with the storage manager. For example, if you use ISM, use the XBSA shared library provided with ISM. ON-Bar and the XBSA shared library must be compiled the same (32-bit or 64-bit).

ON-Bar uses XBSA to exchange the following types of information with a storage manager:

- **Control data.** ON-Bar exchanges control data with a storage manager to verify that ON-Bar and XBSA are compatible, to ensure that objects are restored to the proper instance of the database server and in the proper order, and to track the history of backup objects.
- **Backup or restore data.** During backups and restores, ON-Bar and the storage manager use XBSA to exchange data from specified storage spaces or logical-log files.

ON-Bar uses XBSA transactions to ensure data consistency. All operations included in a transaction are treated as a unit. All operations within a transaction must succeed for objects transferred to the storage manager to be restorable.

The ON-Bar Tables

ON-Bar uses the following catalog tables in the **sysutils** database to track backup and restore operations:

- The **bar_server** table tracks instances of the database server.
- The **bar_object** table tracks backup objects. A *backup object* is a backup of a dbspace, blobspace, sbpace, or logical-log file.
- The **bar_action** table tracks all backup and restore attempts against each backup object, except some log salvage and cold restore events.
- The **bar_instance** table describes each object that is backed up during a successful backup attempt.

For a description of the content of these tables, see [Chapter 10, “ON-Bar Catalog Tables.”](#)

The Emergency Boot Files

The ON-Bar *emergency boot files* reside in the **\$INFORMIXDIR/etc** directory on UNIX and in the **%INFORMIXDIR%\etc** directory on Windows. The emergency boot files contain the information that you need to perform a cold restore and are updated after every backup.

ON-Bar must be able to restore objects from a storage manager even when the tables in the **sysutils** database are not available. During a cold restore, the database server is not available to access **sysutils**, so ON-Bar obtains the information it needs for the cold restore from the emergency boot file.

IDS

Emergency Boot File on Dynamic Server

ON-Bar uses one emergency boot file on Dynamic Server. The filename for the emergency boot file is **ixbar.servernum**, where *servernum* is the value of the **SERVERNUM** configuration parameter.

Emergency Boot Files on Extended Parallel Server

Figure 2-5 lists the types of emergency boot files that Extended Parallel Server uses. Each node with a storage manager contains one backup boot file and one restore boot file. Multiple coservers on a node share a backup boot file and a restore boot file. The database server has one merge boot file.

Use the BAR_BOOT_DIR configuration parameter to specify the location of the emergency boot files. For more information, see “BAR_BOOT_DIR” on page 9-9. If you do not specify BAR_BOOT_DIR, the database server stores the boot files in \$INFORMIXDIR/etc.

Figure 2-5
Emergency Boot Files

Boot File Type	Boot Filename	Description
Backup	Bixbar_hostname.servernum	This file contains backup information and is updated after every backup.
Restore	Rixbar_hostname.servernum	The onbar-merger process re-creates the restore boot files, which the onbar-worker processes use during a cold restore.
Merge	Mixbar_hostname.servernum	The onbar-merger process re-creates the merge boot file during a cold restore. The merge boot file is temporary and is removed when the onbar_d process that created it exits.

During the cold-restore process, ON-Bar follows these steps to create a restore boot file and restore data.

To create the emergency boot files in a cold restore

1. It merges the backup boot files from all coservers and creates a merge boot file for the restore.
2. It distributes the merge boot file to each coserver, renaming it as the restore boot file and overwriting the old restore boot files.
3. It uses the information in the restore boot file instead of the information in the sysutils database to determine which backup copy of each storage space and log to use.

The ON-Bar Activity Log

ON-Bar writes informational, progress, warning, and error messages to the ON-Bar *activity log*. The activity log also records which storage spaces and logical logs were backed up or restored, the progress of the operation, and approximately how long it took. Use the information in the activity log to determine whether a backup or restore operation succeeded. The **ondblog** utility reports its errors in the activity log.

For a list of ON-Bar informational, warning, and error messages, use the **finderr** or **Find Error** utility or view *IBM Informix Error Messages* at <http://www.ibm.com/software/data/informix/pubs/library/>.

Specifying the Location of the Activity Log

For information on how to change the location of the ON-Bar activity log, see [“BAR_ACT_LOG” on page 9-8](#).

Monitoring the Progress of a Backup or Restore

If your backup or restore operations take a long time to complete, knowing the progress is especially useful. Use the `BAR_PROGRESS_FREQ` configuration parameter to specify, in minutes, the frequency of the progress messages written to the ON-Bar activity log. For information on how to change the frequency of the progress messages, see [“BAR_PROGRESS_FREQ” on page 9-16](#).

Configuring the Storage Manager and ON-Bar

In This Chapter	3-3
Configuring a Storage Manager	3-3
Installing and Configuring a Third-Party Storage Manager.	3-3
Configuring ISM	3-4
Updating the sm_versions File	3-5
Configuring Multiple Storage Managers on Coserver Nodes	3-6
Configuring ON-Bar	3-7
Creating the bargroup Group	3-7
Updating the onbar Script	3-8
Setting ISM Environment Variables and ONCONFIG Parameters	3-9
Specifying the Location of the XBSA Library.	3-9
Using ON-Bar Configuration Parameters	3-11
Using ON-Bar Configuration Parameters on Dynamic Server	3-11
Using ON-Bar Configuration Parameters on	
Extended Parallel Server	3-12
Configuring Multiple Storage Managers.	3-12
Global Configuration Parameters	3-13
Storage-Manager Specific Configuration Parameters	3-13
Examples of ON-Bar and Storage-Manager Configurations	3-16
Before You Make a Test Backup	3-19
Choosing Storage Managers and Storage Devices	3-20
Features That ISM Supports	3-20
Features That ISM Does Not Support	3-21
Storage Device Requirements	3-22
Considerations for Extended Parallel Server.	3-23

In This Chapter

This chapter provides the information that you need to plan and set up ON-Bar with a storage manager:

- Installing and configuring the storage manager
- Configuring ON-Bar
- Before you make a test backup
- Choosing storage managers and storage devices

Configuring a Storage Manager

This section discusses installing and configuring a storage manager.

Installing and Configuring a Third-Party Storage Manager

Storage managers have slightly different installation and configuration requirements. Make sure that you follow the manufacturer's instructions carefully. If you have difficulty with the storage-manager installation and configuration, please contact the manufacturer directly. For the list of certified storage managers for your ON-Bar version, consult your sales representative.



Important: Some storage managers let you specify the kind of data to back up to specific storage devices. Configure the storage manager to back up logical logs to one device and storage spaces to a different device for more efficient backups and restores.

To configure a third-party storage manager

1. Set ON-Bar configuration parameters and environment variables.
2. Configure the storage manager so that ON-Bar can communicate correctly with it. For information, see your storage-manager documentation.
3. Configure your storage devices.
To configure your storage devices, follow instructions in your storage-manager documentation. The storage manager must know the device names of the storage devices that it should use.
4. Label your storage volumes.
5. Mount the storage volumes on the storage devices.
6. Update the storage-manager definition in the **sm_versions** file. For more information, see [“Updating the sm_versions File” on page 3-5](#).
7. Verify that the **BAR_BSALIB_PATH** configuration parameter points to the correct XBSA shared library for your storage manager. For more information, see [“Specifying the Location of the XBSA Library” on page 3-9](#).

After you configure the storage manager and storage devices and label volumes for your database server and logical-log backups, you are ready to initiate a backup or restore operation with ON-Bar.

Configuring ISM

For instructions on how to set up ISM to work with ON-Bar, see the *IBM Informix Storage Manager Administrator's Guide*. The ISM server is installed with the Informix database server on UNIX or Windows. Several database server instances can share one ISM instance.



Warning: Install one copy of ISM on each computer to prevent possible conflicts with the XBSA shared library. Do not run ISM and Legato NetWorker on the same computer because they conflict with each other.

Updating the `sm_versions` File

The storage manager must have an entry in the `sm_versions` file. If you are using ISM, put `ism` in the `sm_name` field of `sm_versions`. To find out which code name to use in `sm_versions` for third-party storage managers, see the storage-manager documentation.

The storage-manager definition in the `sm_versions` file uses this format:

```
1|XBSA_ver|sm_name|sm_ver
```

`XBSA_ver` is the release version of the XBSA shared library for the storage manager, `sm_name` is the name of the storage manager, and `sm_ver` is the storage-manager version. The maximum field length is 128 characters.

The following example shows the ISM definition in the `sm_versions` file:

```
1|1.0.1|ism|ISM.2.20.UC1.114|
```

Before ON-Bar starts a backup or restore process, it calls the currently installed version of the storage-manager-specific XBSA shared library to get its version number. If this version is compatible with the current version of ON-Bar and is defined in the `sm_versions` file, ON-Bar begins the requested operation.

To update the storage-manager definition in `sm_versions`

1. Copy the `sm_versions.std` template to a new file, `sm_versions` in the `$INFORMIXDIR/etc` directory on UNIX or the `%INFORMIXDIR%\etc` directory on Windows.
2. If you are using ISM, issue the `ism_startup -init` command to automatically update `sm_versions` with the correct version number and storage-manager name or manually edit `sm_versions`.
If you are installing an ISM patch, you must manually edit `sm_versions`.

Warning: The `ism_startup -init` command erases records of previous backups.



3. If you are using a third-party storage manager, the vendor supplies the definition for the `sm_versions` file. Create your own `sm_versions` file with the correct data for the storage manager using the format in `sm_versions.std` as a template.

XPS

4. If all coservers share the **sm_versions** file in the **etc** subdirectory, the **sm_versions** file should have an entry for each storage-manager brand.

If the **etc** subdirectory is not shared between coserver nodes, specify one line in the **sm_versions** file for the storage manager in use on that coserver node. ♦
5. Stop any ON-Bar processes (**onbar_d**, **onbar_w**, or **onbar_m**) that are currently running and restart them for the changes to take effect.

XPS

Configuring Multiple Storage Managers on Coserver Nodes

Extended Parallel Server allows multiple storage-manager instances, but only one instance per node. You can configure and use different storage-manager brands for different purposes. For best performance, run **onbar-worker** processes on all nodes that have storage devices. For example, you have two storage devices, a tape drive and a jukebox, and want to connect them on different nodes. When an **onbar-worker** is on each node, the data moves faster because it does not have to travel over the network. For complex examples, see [“Using ON-Bar Configuration Parameters on Extended Parallel Server”](#) on page 3-12.

For information on how to update the **sm_versions** file when using multiple storage managers, see step 4 in [“To update the storage-manager definition in sm_versions”](#) on page 3-5.

Some third-party storage managers have a client/server architecture. For these storage managers, *one instance per node* means one storage-manager client.



Important: Each hardware MPP node that contains a coserver that runs **onbar-worker** processes must have a local copy of the storage-manager version of the XBSA shared library.

Validating Your Storage Manager

When you convert or revert an Informix database server, the storage manager that you used on the old version might not be validated for the version that you are migrating to. Verify that the storage-manager vendor has successfully completed the Informix validation process for database server version and platform. If not, you need to install a validated storage manager before you perform backups with ON-Bar.

Configuring ON-Bar

ON-Bar is installed with your Informix database server software. To use ON-Bar with installed storage managers, you set specific parameters in the ONCONFIG file. The following section describes the required ON-Bar configuration parameters.

Use the **onconfig.std** file as a template. ♦

Use the **onconfig.std** file as a template for single coservers. Use the **onconfig.xps** file as a template for multiple coservers. ♦

IDS

XPS

UNIX

Creating the bargroup Group

If you want users other than **informix** or **root** to execute ON-Bar commands, you can create a **bargroup** group. Members of **bargroup** can execute ON-Bar commands. The **bargroup** group on UNIX is similar to the **Informix-Admin** group on Windows. For instructions on how to create a group, see your UNIX documentation.

Important: For security, it is recommended that onbar commands not be run by the **root** user.



Updating the onbar Script

When the installation program installs the database server files, including the ON-Bar files, the **onbar** script is distributed as a shell script so that users can add any preprocessing or postprocessing steps to the script.

To prevent the loss of user changes in the existing **onbar** script, the **onbar** script is distributed as a file named **onbar.sh** (UNIX) or **onbar.bat** (Windows). When the install program installs the database server files over an existing installation, it checks whether any difference exists between the new **onbar** script and the old **onbar** script.

- If the two scripts are the same, the installation program renames the **onbar.sh** or **onbar.bat** file to **onbar**, the new **onbar** script overwrites the old **onbar** script, and no user data is lost.
- If a difference exists between the new **onbar** script and the old **onbar** script, the installation program renames the **onbar.sh** or **onbar.bat** file to **onbar**, renames the old **onbar** script to the form **onbar.date**, and issues a message to the user that the existing **onbar** script was renamed.

If you see a message that the old **onbar** script has been renamed by appending a date, look at the new **onbar** script (filename **onbar**) and integrate the contents of the old **onbar** script into the new **onbar** script. For example, if **onbar** has been renamed to **onbar.2000.12.15**, integrate the contents of **onbar.2000.12.15** into **onbar**.

For information on using the **onbar** script, see [“Customizing ON-Bar and Storage-Manager Commands” on page 8-3](#). For information on installing the database server, see your *Installation Guide*.

Setting ISM Environment Variables and ONCONFIG Parameters

When you use ISM, you need to set certain environment variables. For information, see the *IBM Informix Storage Manager Administrator's Guide*.

IDS

You can set these environment variables in the **onbar** script or in your environment. ♦

XPS

You can set these environment variables in your environment if you start **onbar -w** by hand or before you start the database server, or set them in **start_worker.sh**. ♦

If you use ISM, you can specify the volume pool names for storage spaces and logical logs in the ISM_DATA_POOL and ISM_LOG_POOL parameters in the ONCONFIG file. If you do not set these parameters, they default to the volume pool names ISMData and ISMLogs, respectively.

Specifying the Location of the XBSA Library

UNIX

By default, ON-Bar looks for the XBSA shared library in **\$INFORMIXDIR/lib/libbsa.s[o]** on UNIX. To specify a different name or location of the XBSA shared library, use the **BAR_BSALIB_PATH** configuration parameter. You can also make **\$INFORMIXDIR/lib/ibsad001.s[o]** a symbolic link to the correct library.

For example, if you are using ISM, you can do either of the following:

- Link **\$INFORMIXDIR/lib/ibsad001.so** to **\$INFORMIXDIR/lib/libbsa.so**
- Set **BAR_BSALIB_PATH** to **\$INFORMIXDIR/lib/libbsa.so** ♦

Windows

On Windows, because no default XBSA shared library name exists, you must specify its name and location in the **BAR_BSALIB_PATH** configuration parameter. If you are using ISM, set **BAR_BSALIB_PATH** to **%ISMDIR%\bin\libbsa.dll**. ♦

If you are using a third-party storage manager, ON-Bar must use the version of the XBSA library that the storage-manager manufacturer provides. For more information, see [“BAR_BSALIB_PATH” on page 9-10](#) and your release notes.

XPS

The XBSA library must be present on each coserver node where you are running **onbar-worker** processes. Each **onbar_worker** needs to dynamically link to the functions in the XBSA library. The XBSA library must be either on a local disk or NFS-mounted disk.

To find out whether **onbar_workers** can share libraries and whether the sharing can be done through NFS, check your storage-manager documentation.

You can specify `BAR_BSALIB_PATH` in the global section of the `ONCONFIG` file if you configure:

- The XBSA shared library to have the same path on all nodes
 - Storage managers from more than one vendor if each shared XBSA library has the same path on each node, which is not NFS-mounted
- If each XBSA library uses a different path, you must specify `BAR_BSALIB_PATH` in each storage-manager-specific section of the `ONCONFIG` file. ♦



Important: To set the pathname of the XBSA library with the `BAR_BSALIB_PATH` configuration parameter in the `ONCONFIG` file, specify the absolute pathname. If you specify a relative pathname, then the following message is written to the ON-Bar activity log: `BAR_BSALIB_PATH is ONCONFIG is not an absolute pathname.`

Using ON-Bar Configuration Parameters

Before you begin your first backup, review the default ON-Bar parameters in the ONCONFIG file and adjust the values as needed. For more information, see [“Before You Make a Test Backup” on page 3-19](#). For the complete list of database server configuration parameters and their default values, see the *Administrator’s Reference*.

Using ON-Bar Configuration Parameters on Dynamic Server

ON-Bar on Dynamic Server uses the following configuration parameters.

Configuration Parameter	Reference	Purpose
ALARMPROGRAM	page 9-7	Specifies a script that handles alarms For ON-Bar, set this script to log_full.sh to automatically back up log files when they become full.
BAR_ACT_LOG	page 9-8	Specifies the location and name for the ON-Bar activity log file.
BAR_BSALIB_PATH	page 9-10	Specifies the path of the storage-manager library on UNIX or a dll on Windows To determine if BAR_BSALIB_PATH is supported on your platform, check your release notes.
BAR_HISTORY	page 9-12	Specifies whether the sysutils database maintains the backup history
BAR_MAX_BACKUP	page 9-14	Specifies the maximum number of processes per onbar command
BAR_NB_XPORT_COUNT	page 9-15	Specifies the number of shared-memory data buffers for each onbar_d worker or child process
BAR_PROGRESS_FREQ	page 9-16	Specifies in minutes how frequently the backup or restore progress messages display in the activity log
BAR_RETRY	page 9-17	Specifies how many times ON-Bar should retry a backup, logical-log backup, or restore operation if the first attempt fails

Configuration Parameter	Reference	Purpose
BAR_XFER_BUF_SIZE	page 9-22	Specifies the size in pages of the buffers that the database server uses to exchange data with each onbar_d worker or child process
ISM_DATA_POOL	page 9-24	Specifies the volume pool that you use for backing up storage spaces (ISM)
ISM_LOG_POOL	page 9-25	Specifies the volume pool that you use for backing up logical logs (ISM)
LTAPEDEV	page 9-26	For ontape , specifies the tape device where logical logs are backed up For ON-Bar, specifies whether to back up logs.
RESTARTABLE_RESTORE	page 9-27	Turns restartable restore on or off

(2 of 2)

XPS

Using ON-Bar Configuration Parameters on Extended Parallel Server

The **ONCONFIG** file contains a section for global parameters and individual sections for each storage-manager instance. You might need to specify multiple instances of storage managers to back up and restore data to all the coservers in Extended Parallel Server.

Configuring Multiple Storage Managers

In Extended Parallel Server, you can use more than one storage-manager product, as follows:

- Different versions of a particular storage manager
- Storage managers from different vendors

If you use the **onconfig.std** template to configure a single coserver with one storage manager, copy the section “**Storage-Manager instances**” from **onconfig.xps** into your **ONCONFIG** file. Use the **onconfig.xps** template to configure multiple storage managers.

Warning: *Be careful not to get the shared libraries for the two products or versions mixed up.*



Global Configuration Parameters

The global section includes parameters that apply to all storage managers. You can include ON-Bar parameters in the global section if they are the same for all storage-manager instances.

Put these parameters in the storage-manager section between the BAR_SM and END pair if they are different for each storage-manager instance.

Storage-Manager Specific Configuration Parameters

You must define each storage-manager client that you install and configure in the storage-manager section, as illustrated in [“Defining a Storage Manager on a Five-Coserver System” on page 3-16](#).

Configuration Parameter	Reference	Purpose	Can Be Storage- Manager Specific	Always Storage- Manager Specific	Always Global
BAR_ACT_LOG	page 9-8	Specifies the location and name for the ON-Bar activity log file.			✓
BAR_BOOT_DIR	page 9-9	Specifies the directory for the emergency boot files			✓
BAR_BSALIB_PATH	page 9-10	Specifies the path of the storage-manager library To determine if BAR_BSALIB_PATH is supported on your platform, check your release notes. Specify BAR_BSALIB_PATH in the storage-manager section if the libraries are not in the same location on all nodes.	✓		
BAR_DBS_COSVR	page 9-11	Specifies coservers that send backup and restore data to the storage manager		✓	

(1 of 3)

Configuration Parameter	Reference	Purpose	Can Be Storage- Manager Specific	Always Storage- Manager Specific	Always Global
BAR_HISTORY	page 9-12	Specifies whether the sysutils database maintains a backup history			✓
BAR_IDLE_TIMEOUT	page 9-13	Specifies the maximum number of minutes that an onbar-worker process is idle before it is shut down	✓		
BAR_LOG_COSVR	page 9-13	Specifies coservers that send log backup data to the storage manager		✓	
BAR_PROGRESS_FREQ	page 9-16	Specifies in minutes how frequently the backup or restore progress messages display in the activity log			✓
BAR_RETRY	page 9-17	Specifies how many times ON-Bar should retry a backup, logical-log backup, or restore operation if the first attempt fails			✓
BAR_SM	page 9-19	Specifies the storage-manager number Is required; starts the storage-manager section.		✓	
BAR_SM_NAME	page 9-19	Specifies the storage-manager name		✓	
BAR_WORKER_COSVR	page 9-20	Lists the coservers that can access the storage manager This parameter is required.		✓	

(2 of 3)

Configuration Parameter	Reference	Purpose	Can Be Storage- Manager Specific	Always Storage- Manager Specific	Always Global
BAR_WORKER_MAX	page 9-20	Specifies the maximum number of onbar-worker processes that the Backup Scheduler can start for this storage-manager instance You can start additional onbar-worker processes manually.	✓		
BAR_XFER_BUFSIZE	page 9-23	Specifies the size in pages of the buffers used between XPS and each onbar-worker process			✓
BAR_XPORT_COUNT	page 9-24	Specifies the number of shared-memory data buffers for each onbar-worker process			✓
ISM_DATA_POOL	page 9-24	Specifies the volume pool that you use for backing up storage spaces	✓		
ISM_LOG_POOL	page 9-25	Specifies the volume pool that you use for backing up logical logs	✓		
LOG_BACKUP_MODE	page 9-25	Specifies whether to back up full logical-log files automatically or manually			✓

(3 of 3)

Examples of ON-Bar and Storage-Manager Configurations

This section shows sample storage-manager configurations for Extended Parallel Server. For more information about each configuration parameter, refer to [Chapter 9, “Setting ON-Bar Configuration Parameters.”](#)

Creating a Storage-Manager Definition

The following configuration example is for a storage manager that can run on coservers 1, 2, 3, 4, and 7. In this configuration, you have to start **onbar-worker** processes manually on coservers 1, 2, 3, 4, and 7, because `BAR_WORKER_MAX` is not set. All storage spaces and logical logs are backed up to this storage-manager instance.

```
# Backup/Restore Variables
BAR_ACT_LOG      /tmp/bar_act.log    # Path of activity log
BAR_RETRY        2                  # Number of times to retry failures
BAR_XPORT_COUNT  10                 # Number of transport buffers per worker
BAR_XFER_BUFSIZE 8                  # Size of each transport buffer in pages
LOG_BACKUP_MODE  CONT               # Backup as soon as logical log fills
BAR_IDLE_TIMEOUT 5                  # How long onbar-workers wait
BAR_BSA_LIB_PATH /usr/lib/ibsad001.so # XBSA shared lib path

# Storage-Manager Section
BAR_SM 1
BAR_WORKER_COSVR 1-4,7
END
```

Defining a Storage Manager on a Five-Coserver System

The following example is a simple storage-manager definition that automatically starts a single **onbar-worker** process on coserver 1. Data on coservers 1 through 5 is backed up or restored to the storage manager on coserver 1. If you omit the `BAR_WORKER_MAX` parameter, you must start **onbar-worker** processes manually. For more information, see [“Using start_worker.sh to Start onbar_worker Processes Manually”](#) on page 8-7.

```
# Storage Manager instances
BAR_SM 1                  # Storage manager ID
  BAR_SM_NAME A           # Storage manager name
  BAR_WORKER_COSVR 1      # Storage mgr is on coserver 1
  BAR_DBS_COSVR 1-5       # Route dbspaces to this storage mgr
  BAR_LOG_COSVR 1-5       # Route logs to this storage mgr
  BAR_WORKER_MAX 1        # Number of onbar-workers
END
```

Defining the Number of onbar-worker Processes on Two Storage Managers

The following example defines different storage managers on two coservers. Because the global BAR_WORKER_MAX value is 3, the Backup Scheduler will start up to 3 **onbar-worker** processes on coserver 2 for storage-manager **BAKER**. For storage-manager **ABEL**, the local BAR_WORKER_MAX value overrides the global setting, so the Backup Scheduler will start only one **onbar-worker** process.

```
# Global section
BAR_WORKER_MAX      3      # Global value for no. of onbar-workers

# Storage Manager ABEL
BAR_SM      1
  BAR_WORKER_MAX      1 # only one onbar-worker defined
  BAR_DBS_COSVR      1
  BAR_LOG_COSVR      1
  BAR_WORKER_COSVR      1
END

# Storage Manager BAKER
BAR_SM      2
  BAR_DBS_COSVR      2
  BAR_LOG_COSVR      2
  BAR_WORKER_COSVR      2
END
```



Important: Do not switch the BAR_SM identification numbers between different storage managers when you reconfigure the storage managers, or else ON-Bar cannot find the backup objects. For example, do not reassign BAR_SM 1 to storage-manager **BAKER** and BAR_SM 2 to storage-manager **ABEL**.

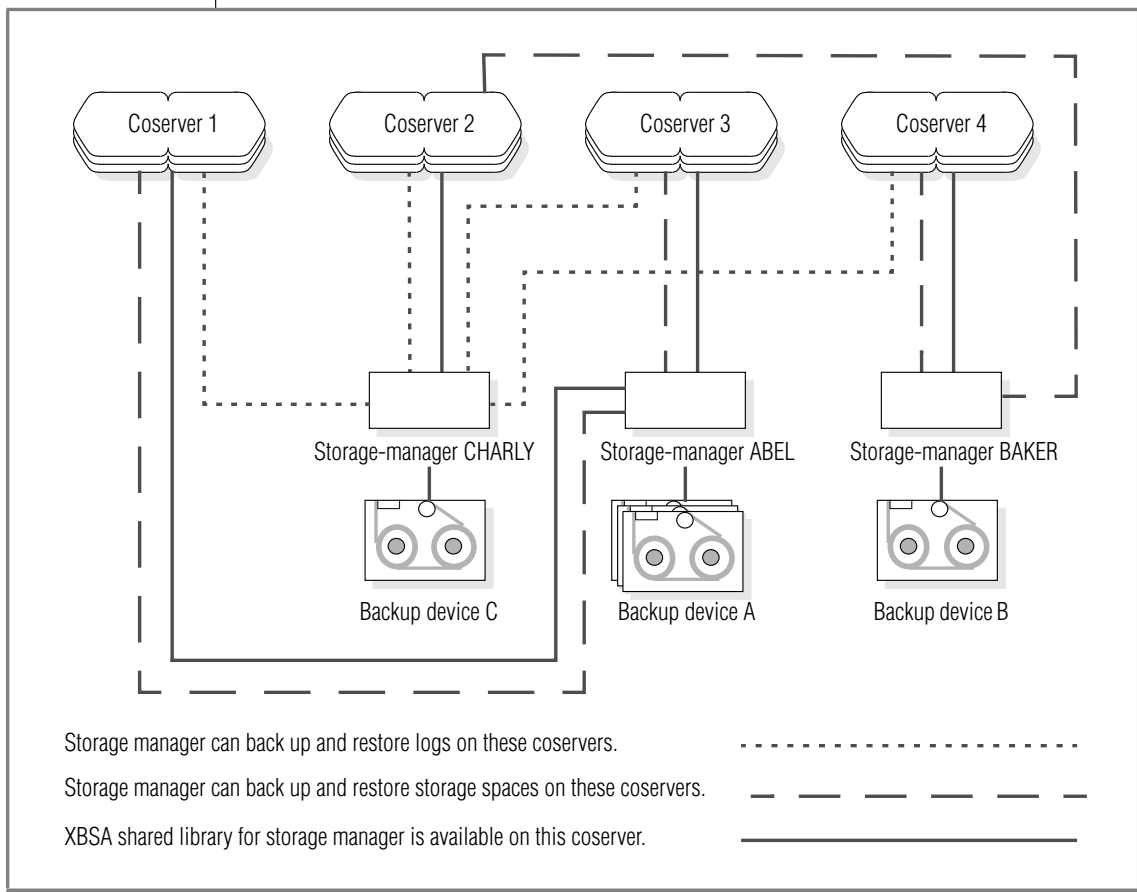
Defining Three Storage Managers and Storage Devices

The configuration in [Figure 3-1 on page 3-18](#) shows how you might set up three storage managers and three backup devices:

- A, a silo with two drives and two connections, one to coserver 1 and the other to coserver 3
- B, a tape autochanger connected to coserver 4
- C, a simple tape drive connected to coserver 2

Storage-manager **ABEL** can back up and restore storage spaces on coservers 1 and 3. Storage-manager **BAKER** can back up and restore storage spaces on coservers 4 and 2. Storage-manager **CHARLY** can back up and restore logs on all four coservers.

Figure 3-1
Storage-Manager Configuration



The ONCONFIG definitions for storage-managers **ABEL**, **BAKER**, and **CHARLY** appear in the following example:

```
# Storage manager section for storage manager A
BAR_SM 1
BAR_SM_NAME ABEL
BAR_WORKER_COSVR 1,3
BAR_DBS_COSVR 1,3
BAR_LOG_COSVR 0
BAR_WORKER_MAX 2
END
# Storage manager section for storage manager B
BAR_SM 2
BAR_SM_NAME BAKER
BAR_WORKER_COSVR 4
BAR_DBS_COSVR 2,4
BAR_LOG_COSVR 0
BAR_WORKER_MAX 1
END
# Storage manager section for storage manager C
BAR_SM 3
BAR_SM_NAME CHARLY
BAR_WORKER_COSVR 2
BAR_DBS_COSVR 0
BAR_LOG_COSVR 1 - 4
BAR_WORKER_MAX 1
END
```

Before You Make a Test Backup

Check the items in the following list to make sure that ON-Bar and your storage manager are set up correctly:

- The storage manager is installed and configured to manage specific storage devices.
- Make sure that the `BAR_BSALIB_PATH` configuration parameter specifies correctly the XBSA shared library or it is not set and the library is in the default location. ♦
- Make sure that the `BAR_BSALIB_PATH` configuration parameter specifies correctly the XBSA shared library. ♦
- The `sm_versions` file contains a row that identifies the version number of the storage-manager-specific XBSA shared library.
- The `BAR_WORKER_MAX` parameter is set to a number greater than 0 in the storage-manager-specific section of the ONCONFIG file. ♦

UNIX

Windows

XPS

After you verify that ON-Bar and your storage manager are set up correctly, run ON-Bar on your test database to make sure that you can back up and restore data. For more information, follow the instructions in [Chapter 4, “Backing Up with ON-Bar.”](#)

Choosing Storage Managers and Storage Devices

The storage manager manages the storage devices to which the backed-up data is written. ISM is included with your database server. For information on how to use ISM, refer to the *IBM Informix Storage Manager Administrator's Guide*.

If you choose a different storage manager, consider whether it has the features that you need to back up your storage spaces and logical logs. When you choose storage devices, make sure that they are compatible with the storage manager that you choose. The storage devices should have the speed and capacity that your backups require. The storage manager should be easy to use and work on your operating system.

Features That ISM Supports

ISM fulfills the following storage-manager requirements:

- ISM allows you to back up logical logs and storage spaces to different devices and to specify whether to use encryption or compression for data.
- ISM can write the output of parallel backups to a single device, medium, or volume. Some backup devices can write data faster than the disks used to hold storage spaces can be read.
- ISM can automatically switch from one tape device to another when the volume in the first device fills.
- ISM allows migration of data from one backup medium to another. For speed, you can back up logical logs or storage spaces to disk, but you must move them later to tape or other removable media or your disk will become full.

- ISM allows you to clone copies of backup data for on-site and off-site storage.
- ISM uses automatic expiration of data. Once all data on a backup media expires, you can reuse the media.

Features That ISM Does Not Support

ISM does not support the following features. Third-party storage managers might support these features.

- Distributing a single data stream across multiple devices simultaneously, which improves throughput if you have several slow devices
- Using different encryption or compression methods for specified storage spaces or databases
- Scheduling backups
- Support for devices such as tape libraries, jukeboxes, silos, tape autochangers, and stackers
- Remote host operations

You can install some storage managers on a different host from the database server. However, ISM must be installed on the same host as the database server.

Storage Device Requirements

Ask the following interrelated questions to determine what storage devices you need. For example, the speed and type of storage devices partly determine the number of storage devices that you need.

- What kind of storage devices do you need?

The transaction volume and the size of your database are major factors in determining the kind of storage devices that you need.

ISM supports simple tape devices such as QIC, 4mm, 8mm, DLT, optical devices, and disk backups. If ISM cannot manage the storage devices that you need, you need to purchase a different storage manager. For more information, see the *IBM Informix Storage Manager Administrator's Guide*.

- What is the availability requirement for each device?

Is it important for your storage devices to allow random as well as sequential access? If so, you cannot use tape storage devices.

- How many storage devices do you need?

ISM supports up to four devices per host. The number of storage devices that you need depends on the kind of storage devices you have, how much transaction activity occurs on the database server, how fast throughput is, how much time you can allow for backups, and other similar factors.

- How many and what type of storage-manager instances should you configure?

You can have one storage manager on each coserver node. ♦

XPS

Considerations for Extended Parallel Server

Because backing up data on multiple coservers is much more complex, some Extended Parallel Server users use third-party storage managers. The following usage requirements also affect your decisions about the storage manager and storage devices:

- The number of hardware nodes, the number of coservers on those nodes, and how the coservers are distributed across the nodes
Balance these factors against the number of storage devices and storage-manager instances. The architecture of some platforms limits where you can attach devices, but the number of coservers increases processing requirements. The storage-manager sections of the ONCONFIG file should reflect these considerations.
Although some nodes in a massively parallel processing (MPP) system might not be running coservers, they might be able to run part of the storage manager.
- The kind of high-speed interconnect on the MPP system
Because disks are slower than the high-speed interconnect, they could create a bottleneck in the interconnect. Distributing devices across multiple nodes might reduce the amount of traffic across the interconnect and allow more parallelism.
- If you isolate tables or databases in a single storage space or in a dbslice across coservers, you can restore single tables or databases.
- Whether it is possible to restore data from external sources
Although Decision Support System (DSS) databases might not be mirrored, they might be easier to re-create from the original external source than to restore from backups if they are corrupted or damaged. For Online Transaction Processing (OLTP) systems, regenerating data from external sources is rarely possible.

- The size of each logical-log stream, how the transaction activity is distributed across logical-log streams, and when it occurs

If you are running an OLTP system with many transactions evenly distributed across coservers, your storage manager and storage-device requirements are different from a DSS, which usually generates few transactions.

In addition, if logstreams are the same size on each coserver but activity is not evenly distributed, space and resources are wasted. You should adjust them for efficiency.

Backing Up with ON-Bar

In This Chapter	4-3
Syntax of ON-Bar Commands	4-4
Preparing for a Backup	4-5
What Data Does ON-Bar Back Up?	4-5
Which Administrative Files to Back Up?	4-6
Installing and Configuring a Storage Manager	4-7
What Is a Whole-System Backup?	4-7
What Is a Standard Backup?	4-7
What Is a Physical Backup?	4-7
Choosing a Backup Level	4-8
Level-0 Backups	4-8
Level-1 Backups	4-8
Level-2 Backups	4-8
Collecting Information About Your System Before a Backup	4-9
Ensuring That You Have Enough Logical-Log Space	4-9
Copying Database Server Configuration Information	4-9
Verifying Database Integrity	4-10
Backing Up Storage Spaces and Logical Logs	4-10
Backup Syntax	4-11
Backing Up After Changing the Physical Schema	4-13
When to Back Up the Root Dbspace and Modified Storage Spaces	4-13
When to Back Up the Modified Storage Spaces Only	4-14
Using ISM During a Backup	4-14
Using IBM Informix Server Administrator to Back Up and Verify	4-15
ON-Bar Backup Examples	4-16
Performing a Level-0 Backup of All Storage Spaces	4-16
Performing a Level-0 Backup of Specified Storage Spaces	4-16

Performing an Incremental Backup.	4-16
Backing Up a List of Storage Spaces Specified in a File	4-16
Backing Up Specific Tables	4-17
Retrying Skipped Storage Spaces During a Backup	4-18
Performing a Whole-System Backup	4-18
Backing Up Smart Large Objects in Sbspaces	4-18
Using Fake Backups in a Data Warehouse	4-19
Backing Up Blobspaces in a Logging Database.	4-19
Backing Up Logical Logs When Blobspaces Are Offline.	4-20
Assigning a Name to a Backup Session	4-20
Performing a Physical Backup	4-20
Backing Up a Dbslice	4-20
Backing Up Table Types	4-21
Backing Up Logical Logs	4-21
Backing Up Logical Logs on Dynamic Server	4-22
Performing a Continuous Backup of Logical Logs	4-23
Performing a Manual Backup of Logical Logs	4-24
Using ALARMPROGRAM to Set the Log Backup Mode	4-24
Backing Up Logical Logs on Extended Parallel Server	4-25
Performing a Manual Backup of Logical Logs	4-26
Starting Continuous Logical-Log Backups	4-27
Preventing Logical-Log Backups in a Test System.	4-27
Monitoring Logical-Log Backups.	4-27
Salvaging Logical-Log Files.	4-28
Skipping Logical Replay.	4-29
Restoring Nonlogging Databases and Tables.	4-29
Understanding ON-Bar Backup Processes	4-30
Backup Sequence on Dynamic Server	4-30
Backup Sequence on Extended Parallel Server	4-32

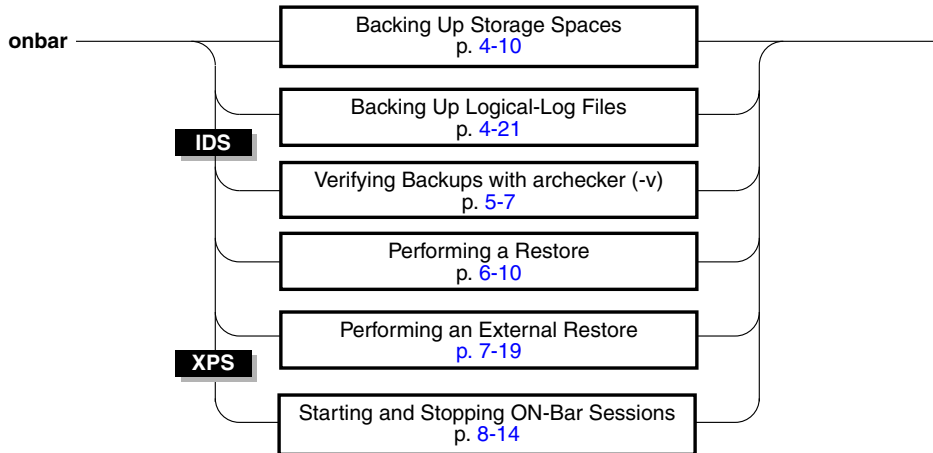
In This Chapter

This chapter explains how to use the **onbar** utility to back up and verify storage spaces (dbspaces, blobspaces, and sbspaces) and logical-log files. The **onbar** utility is a wrapper to **onbar_d**, the ON-Bar driver. Use any of the following methods to execute ON-Bar backup and restore commands:

- Issue ON-Bar commands.
To execute ON-Bar commands, you must be user **informix** or a member of the **bargroup** group on UNIX or a member of the **Informix-Admin** group or user **informix** in Windows. (For more information, see [“Creating the bargroup Group” on page 3-7.](#))
- Include ON-Bar and ISM commands in a shell or batch script.
For information, see [“Customizing ON-Bar and Storage-Manager Commands” on page 8-3.](#)
- Call ON-Bar from a job-scheduling program.
- Set event alarms that trigger a logical-log backup.
For information, see [“Backing Up Logical Logs on Dynamic Server” on page 4-22](#) and [“Backing Up Logical Logs on Extended Parallel Server” on page 4-25.](#)

Syntax of ON-Bar Commands

You can use ON-Bar to back up and restore storage spaces and logical logs, to verify a backup, and to start or stop ON-Bar sessions.



Preparing for a Backup

This section explains the preliminary steps that you must take before you back up storage spaces and logical logs.

What Data Does ON-Bar Back Up?

ON-Bar backs up the following types of data. ON-Bar backs up the critical dbspaces first, then the remaining storage spaces, and finally the logical logs. (The *critical dbspaces* are the **rootdbs** and the dbspaces that contain the logical logs and physical log.) ON-Bar can back up and restore the largest storage space that your database server supports.

Data Type	Description
Dbspaces that contain tables or indexes	See “Backing Up Storage Spaces and Logical Logs” on page 4-10 . ON-Bar also backs up the reserved pages in the root dspace.
Blobspaces (IDS)	See “Backing Up Blobspaces in a Logging Database” on page 4-19 .
ISM catalog	If you use ISM, the ISM catalog is in \$INFORMIXDIR/ism on UNIX and %ISMDIR% on Windows.
Logical-log files	See “Backing Up Logical Logs” on page 4-21 .
Sbspaces (IDS)	See “Backing Up Smart Large Objects in Sbspaces” on page 4-18 .



Which Administrative Files to Back Up?

ON-Bar backups safeguard your data. They do not replace normal operating-system backups of important configuration files.

Important: For use in an emergency, you should have a backup copy of the current version of the following administrative files. You will need to restore these files if you need to replace disks or if you restore to a second computer system (imported restore).

The following table lists the administrative files that you should back up.

Administrative Files	IDS	XPS
ONCONFIG file	✓	✓
Emergency boot files	✓	✓
sm_versions file	✓	✓
The sqlhosts file (UNIX)	✓	✓
The oncfg_servername.servernum file in the etc subdirectory	✓	
The oncfg_servername.servernum.coserverid file from each coserver		✓
Storage-manager configuration and data files	✓	✓
Simple-large-object data in blobspaces that are stored on disks or optical platters	✓	
The xcfg_servername.servernum file in the etc subdirectory		✓
Externally stored data such as external tables that a DataBlade maintains	✓	

Although ON-Bar does not back up the following items, ON-Bar automatically re-creates them during a restore. You do not need to make backup copies of these files:

- The *dbspace* pages that are allocated to the database server but that are not yet allocated to a *tblspace* extent
- Mirror chunks, if the corresponding primary chunks are accessible
- Temporary *dbspaces*

ON-Bar does not back up or restore the data in temporary *dbspaces*. Upon restore, the database server re-creates empty temporary *dbspaces*.

Installing and Configuring a Storage Manager

Before you can create a backup with ON-Bar, you must configure your storage manager and start it. For information about configuring ISM, see the *IBM Informix Storage Manager Administrator's Guide*. For information about configuring third-party storage managers, see [Chapter 3, “Configuring the Storage Manager and ON-Bar,”](#) and your storage-manager manuals.

Make sure your storage manager is ready to receive data before you begin a backup or restore. To improve performance, It is recommended that you reserve separate storage devices for storage-space and logical-log backups. Label and mount all volumes in the storage device. The backup or restore might pause until you mount the requested tape or optical disk.

IDS

What Is a Whole-System Backup?

A *whole-system* backup (**onbar -b -w**) is a serial backup of all storage spaces and logical logs based on a single checkpoint. That time is stored with the backup information. The advantage of using a whole-system backup is that you can restore the storage spaces with or without the logical logs. Because the data in all storage spaces is consistent in a whole-system backup, you do not need to restore the logical logs to make the data consistent. For an example, see [“Performing a Whole-System Backup” on page 4-18.](#)

What Is a Standard Backup?

A *standard backup* (**onbar -b**) is a parallel backup of selected or all storage spaces and the logical logs. In a standard backup, the database server performs a checkpoint for each storage space as it is backed up. Therefore, you must restore the logical logs from a standard backup to make the data consistent. For an example, see [“Performing a Level-0 Backup of All Storage Spaces” on page 4-16.](#)

XPS

What Is a Physical Backup?

A *physical backup* (**onbar -b -p**) backs up just the storage spaces. You can back up specific or all storage spaces.



Choosing a Backup Level

ON-Bar supports level-0, level-1, and level-2 backups.

Tip: *It is good practice to create a backup schedule that keeps level-1 and level-2 backups small and to schedule frequent level-0 backups. With such a backup schedule, you avoid having to restore large level-1 and level-2 backups or many logical-log backups.*

Level-0 Backups

Level-0 backups can be time consuming because ON-Bar writes all the disk pages to backup media. Level-1 and level-2 backups might take almost as much time as a level-0 backup because the database server must scan all the data to determine what has changed since the last backup. It takes less time to restore data from level-0, level-1, and level-2 backups than from level-0 backups and a long series of logical-log backups.

Level-1 Backups

A level-1 backup takes less space and might take less time than a level-0 backup because only data that changed since the last level-0 backup is copied to the storage manager.

If you request an incremental backup where no previous incremental backup exists, ON-Bar automatically performs the lower-level backup. For example, if you request a level-1 backup but no level-0 backup exists for one of the dbspaces, ON-Bar automatically performs a level-0 backup of that dbspace and a level-1 backup of the other storage spaces.

If you request a whole-system level-1 backup and no level-0 backup exists, ON-Bar performs a whole-system level-0 backup. If you request a whole-system level-2 backup but the level-1 backup does not exist, ON-Bar performs a whole-system level-1 backup. ♦

Level-2 Backups

A level-2 backup takes less space and might take less time than a level-1 backup because only data that changed since the last level-1 backup is copied to the storage manager.

Collecting Information About Your System Before a Backup

To ensure that you can restore the data, perform the following tasks:

- Print or keep a copy of essential database server configuration information.
- Verify data consistency.
- Keep track of the number of rows in each table (optional).

After you complete the backup, verify it with the **archecker** utility. For more information, see [Chapter 5, “Verifying Backups.”](#)

Ensuring That You Have Enough Logical-Log Space

ON-Bar checks for available logical-log space at the beginning of a backup. If the logs are nearly full, ON-Bar backs up and frees the logs before attempting to back up the storage spaces. If the logs contain ample space, ON-Bar backs up the storage spaces, then the logical logs.

Monitor the logs so that you can back them up before they fill. If insufficient space exists in the logical log, the database server will hang. If the database server hangs, add more logical-log files and retry the ON-Bar command.

Extended Parallel Server keeps one logical log reserved for backups. That way, enough log space exists for you to back up the logical logs to free enough log space to back up storage spaces. ♦

Copying Database Server Configuration Information

Copy the following database server configuration files. For more information, see [“Which Administrative Files to Back Up?”](#) on page 4-6.

- The **sqlhosts** file (UNIX only)
- The **oncfg** files
- The emergency boot files
- The **ONCONFIG** file
- **sm_versions** file
- The **xcfg** file for the database server ♦

XPS

XPS

Verifying Database Integrity

To ensure the integrity of your backups, periodically verify that all database server data is consistent before you create a level-0 backup. You do not need to check for consistency before every level-0 backup. It is recommended that you do not discard a backup that is known to be consistent until the next time that you verify the consistency of your databases. For information on using the **oncheck** or **onutil** CHECK commands, see the *Administrator's Reference*.

Backing Up Storage Spaces and Logical Logs

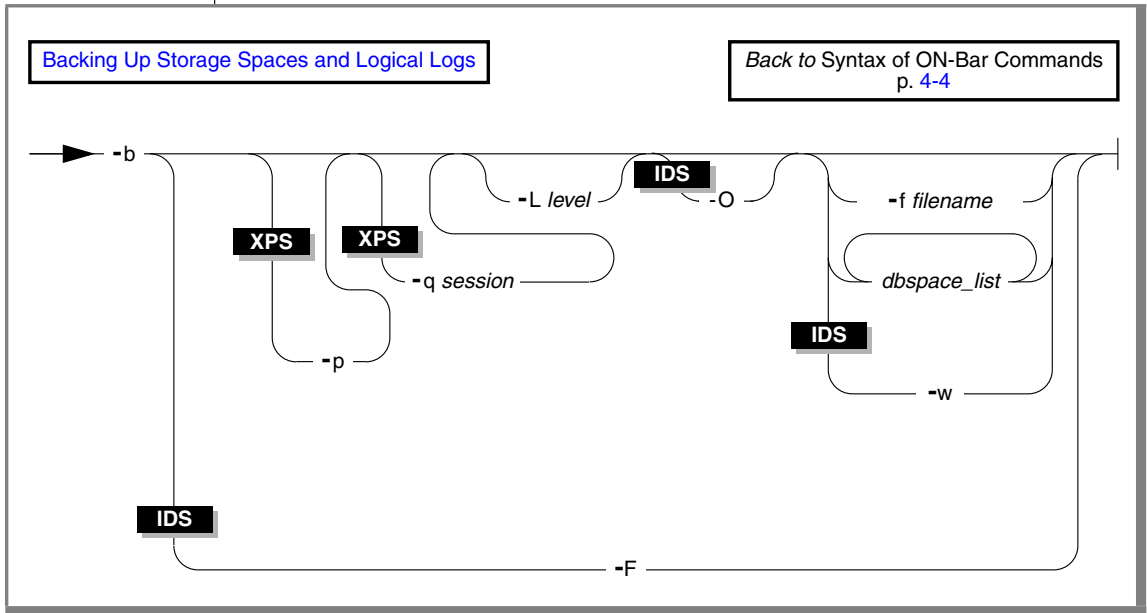
You can back up storage spaces and logical logs only when the database server is in online, quiescent, or fast-recovery mode. However, you can salvage logical logs with the database server offline. The storage-space chunks can be stored on raw disk storage space, in cooked files, or on an NTFS file system (Windows).

Only online storage spaces are backed up. Use the **onstat -d** utility to determine which storage spaces are online. After you begin the backup, monitor its progress in the ON-Bar activity log and database server message log.



Important: *You must back up each storage space at least once. ON-Bar cannot restore storage spaces that it has never backed up.*

Backup Syntax



Element	Purpose	Key Considerations
-b	Specifies a backup Backs up the storage spaces, logical logs, including the current logical log, and the ISM catalog, if it exists.	You must specify the -b parameter first. Important: During a backup, if ON-Bar encounters a down dbspace, it skips it and later returns an error.
<i>dbspace_list</i>	Names storage spaces to be backed up On XPS, if you enter a dbslice name, it backs up all the dbspaces in that dbslice.	If you do not enter <i>dbspace_list</i> or -f filename , ON-Bar backs up all online storage spaces on the database server. If you enter more than one storage-space name, use a space to separate the names.

(1 of 3)

Element	Purpose	Key Considerations
-f filename	<p>Backs up the storage spaces that are listed in the text file whose pathname <i>filename</i> provides</p> <p>Use this option to avoid entering a long list of storage spaces every time that you back up.</p>	<p>The filename can be any valid UNIX or Windows filename, including simple (listfile_1), relative (../backup_lists/listfile_2 or ..\backup_lists\listfile2), and absolute (/usr/informix/backup_lists/listfile3 or c:\informix\backup_lists\listfile3) filenames.</p> <p>For the format of this file, see Figure 4-1 on page 4-17. The file can list multiple storage spaces per line.</p>
-F	Performs a fake backup (IDS)	<p>You can execute this option whether or not a storage-manager application is running. ON-Bar ignores <i>dbspace_list</i> if you specify it. Use fake backups to change database logging modes; to allow the user to use new logs, chunks, or mirrors without performing a backup; or in special situations when you, the administrator, judge that a backup is not needed. No backup actually occurs, so no restore is possible from a fake backup. It is recommended that you use fake backups sparingly, if at all.</p>
-L level	<p>Specifies the level of backup to perform on storage spaces:</p> <ul style="list-style-type: none"> ■ 0 for a complete backup (The default.) ■ 1 for changes since the last level-0 backup ■ 2 for changes since the last level-1 backup 	<p>If you request an incremental backup and ON-Bar finds that no previous level backup has been performed for a particular storage space, ON-Bar backs up that storage space at the previous level.</p> <p>For example, if you request a level-1 backup, and ON-Bar finds no level-0 backup, it makes a level-0 backup instead.</p> <p>For more information, see “Performing an Incremental Backup” on page 4-16.</p>
-O	Overrides normal backup restrictions	<p>Use this option to back up logical logs when blobspaces are offline.</p> <p>If a log backup occurs when blobspaces are offline, return code 178 displays in the ON-Bar activity log.</p>
-p	Performs a physical backup (XPS)	Backs up storage spaces and the ISM catalog. (If you omit the -p option, ON-Bar also backs up the logical logs.)

(2 of 3)

Element	Purpose	Key Considerations
-q session	Allows you to assign a name to the backup session (XPS)	<DBSERVERNAME><random_number> is the default session name. The session name must be unique and can be up to 128 characters. This name appears in the output from the onstat utility so that you can follow the progress of the backup. For more information, see “Monitoring the Backup Scheduler Status” on page 8-17 .
-w	Performs a whole-system backup (IDS)	Backs up all storage spaces, critical dbspaces, and logical logs serially. If you do not save the logical logs, you must use the -w option.

(3 of 3)

Backing Up After Changing the Physical Schema

This section describes what to back up after you change the physical schema.

When to Back Up the Root Dbspace and Modified Storage Spaces

You must perform a level-0 backup of the root dbspace and the modified storage spaces to ensure that you can restore the data after you:

- Add or drop mirroring.
- Move, drop, or resize a logical-log file.
- Change the size or location of the physical log.
- Change your storage-manager configuration.
- Add, move, or drop a dbspace.
- Add, move, or drop a chunk to any type of storage space.
- Add, move, or drop a blobspace or sbpace. ♦
- Add or drop a dbslice or logslice. ♦

For example, if you add a new dbspace **dbs1**, you see a warning in the message log that asks you to perform a level-0 backup of the root dbspace and the new dbspace. If you attempt an incremental backup of the root dbspace or the new dbspace instead, ON-Bar automatically performs a level-0 backup of the new dbspace.

IDS

XPS

XPS



For Extended Parallel Server, back up the root dbspace and the new or modified dbspace on the coserver where the change occurred. ♦

Tip: Although you no longer need to backup immediately after adding a log file, your next backup should be level-0 because the data structures have changed.

Warning: If you create a new storage space with the same name as a deleted storage space, perform a level-0 backup twice:

1. Back up the root dbspace after you drop the storage space and before you create the new storage space with the same name.
2. After you create the new storage space, back up the root dbspace and the new storage space.

When to Back Up the Modified Storage Spaces Only

You must perform a level-0 backup of the modified storage spaces to ensure that you can restore the data when you:

- Convert a nonlogging database to a logging database.
- Convert a raw, static, or operational table to standard. This backup ensures that the unlogged data is restorable before you switch to a logging table type.

Using ISM During a Backup

Use the **ism_watch** command to monitor backups and restores sent to the ISM server. During a backup, the ISM server automatically routes storage-space data to volumes in the ISMData volume pool and logical-log files to volumes in the ISMLogs volume pool.

Always keep the volumes from the ISMLogs pool mounted to ensure that a storage device is always available to accept logical-log data. If the volume is not mounted, the backup will pause. For more information on using devices and ISM commands, see the *IBM Informix Storage Manager Administrator's Guide*.

During the backup operation, ISM creates *save sets* of the backed up data and enters information about the backed up data in the ISM catalog. You also can use this command to back up the ISM catalog:

```
ism_catalog -create_bootstrap
```

If you use the **onbar** script to back up storage spaces and logical logs, it backs up the ISM catalog automatically. If you call **onbar_d** directly, you must use the **ism_catalog -create_bootstrap** command.

Using IBM Informix Server Administrator to Back Up and Verify

IBM Informix Server Administrator (ISA) is a browser-based tool for performing administrative tasks such as ON-Bar and **onstat** commands. You can use ISA to perform the following ON-Bar tasks:

- View messages in the ON-Bar activity log.
- Perform level-0, level-1, or level-2 backups.
 - Back up storage spaces (**onbar -b**).
 - Back up the whole system (**onbar -b -w**).
 - Override error checks during the backup (**onbar -b -O**).
 - Perform a fake backup (**onbar -b -F**).
- Back up the logical logs.
 - Include the current log in the log backup (**onbar -b -l -c**).
 - Override error checks to back up logs when blobspaces are offline (**onbar -l -O**).
 - Start a continuous logical-log backup (**onbar -b -l -C**).
- Verify backups.
- Edit the **onbar** script.

For more information, see the ISA online help.

ON-Bar Backup Examples

The following sections contain examples of ON-Bar syntax for backing up storage spaces.

Performing a Level-0 Backup of All Storage Spaces

To perform a standard, level-0 backup of all online storage spaces and used logical logs, use one of the following commands:

```
onbar -b
onbar -b -L 0
```

ON-Bar never backs up offline storage spaces, temporary dbspaces, or temporary sbspaces.

Important: Save your logical logs so that you can restore from this backup.

Performing a Level-0 Backup of Specified Storage Spaces

To perform a level-0 backup of specified storage spaces and all logical logs (for example, two dbspaces named fin_dbspace1 and fin_dbspace2), use the -b option as the following example shows. You could also specify the -L 0 option, but it is not necessary.

```
onbar -b fin_dbspace1 fin_dbspace2
```

Performing an Incremental Backup

An incremental backup backs up all changes in the storage spaces since the last level-0 backup and performs a level-0 backup of used logical logs. To perform a level-1 backup, use the -L 1 option, as the following example shows:

```
onbar -b -L 1
```

Backing Up a List of Storage Spaces Specified in a File

To back up a list of storage spaces specified in a file and the logical logs, use the following command:

```
onbar -b -f /usr/informix/backup_list/listfile3
```



The format of the file is as follows:

- Each line can list more than one storage space, separated by spaces or a tab.
- Comments begin with a # or ; symbol and continue to the end of the current line.
- ON-Bar ignores all comment or blank lines in the file.
- ON-Bar truncates pathnames to the word after the last directory delimiter.

Figure 4-1 shows a sample file that contains a list of storage spaces to be backed up (**blobsp2.1**, **my_dbSPACE1**, **blobsp2.2**, **dbsl.1**, **rootdbs.1**, and **dbsl.2**). You can also use this file to specify a list of storage spaces to be restored.

Figure 4-1
Sample File with a List of Storage Spaces

```
blobsp2.1
# a comment                                ignore this text

        /usr/informix/my_dbSPACE1           # back up this dbSPACE
; another comment
blobsp2.2                                /usr/informix/dbsl.1
/usr/informix/rootdbs.1  dbsl.2  ; backing up two spaces
```

Backing Up Specific Tables

To back up a specific table or set of tables in ON-Bar, store these tables in a separate dbSPACE and then back up this dbSPACE.

Warning: If you need to restore only that table, you must warm restore the entire dbSPACE to the current time (**onbar -r**). This procedure does not allow you to recover from accidentally dropping or corrupting a table because it would be dropped again during logical restore.



IDS

Retrying Skipped Storage Spaces During a Backup

You cannot back up storage spaces that are down or temporarily inaccessible. If a storage space is down, ON-Bar skips it during the backup and writes a message to the activity log. Take one of the following actions:

- Retry the backup later when the storage space is back online.
- Restore these storage spaces from an older backup, if available. Make sure that at least one level-0 backup of each storage space exists or else it might not be restorable. For details, see [“Restoring from an Older Backup”](#) on page 6-22.

Performing a Whole-System Backup

To perform a serial, level-0 backup of all online storage spaces and logical logs, use one of the following commands:

```
onbar -b -w
onbar -b -w -L 0
```

For more details, see [“Performing a Whole-System Restore”](#) on page 6-21.

IDS

Backing Up Smart Large Objects in Sbspaces

You can back up smart large objects in one or more sbspaces or include them in a whole-system backup. In a level-0 backup, the entire sbpace is backed up. In a level-1 or level-2 backup, the modified sbpages in the sbpace are backed up.

The following example performs a level-0 backup of the **s9sbpace** sbpace:

```
onbar -b -L 0 s9sbpace
```

When you turn on logging for a smart large object, you must immediately perform a level-0 backup to ensure that the object is fully recoverable. For details on logging sbspaces, see the *Administrator’s Guide*.

IDS



IDS



Using Fake Backups in a Data Warehouse

The High-Performance Loader (HPL) in Express mode loads tables in read-only mode. A backup changes the table to update mode. Use one of the following commands:

```
onbar -b          # the recommended way
onbar -b -F
```

Important: *It is recommended that you use fake backups on test systems, not on production systems. You cannot restore data from a fake backup. (If you performed a fake backup, you must reload the table to be able to restore the data.)*

Backing Up Blobspaces in a Logging Database

Follow these steps when you back up blobspaces in a database that uses transaction logging:

1. Before you back up a new blobspace, make sure that the log file that recorded the creation of the blobspace is no longer the current log file.

To verify the logical-log status, use the **onstat -l** or **xctl onstat -l** command. To switch to the next log file, use the **onmode -l** command.

Blobspaces are not available for use until the log file is not the current log file. For more information on verifying the logical-log status, see “[onstat -l](#)” on page B-6. For information on switching log files, see the **onmode** section in the *Administrator's Reference*.

2. If you update or delete simple large objects in a blobspace, you must back up all the log files, including the current log file. If the blobspace is online, use the **onbar -b -l -c** command.

When users update or delete simple large objects in blobspaces, the blobpages are not freed for reuse until the log file that contains the delete records is freed. To free the log file, you must back it up.

3. Back up the blobspaces with the **onbar -b** or **onbar -b -w** command.

Warning: *If you perform a warm restore of a blobspace without backing up the logical logs after updating or deleting data in it, that blobspace might not be restorable.*

IDS



Backing Up Logical Logs When Blobspaces Are Offline

To back up the logical logs when a blobspace is offline, use the **onbar -b -l -O** or **onbar -b -O** command. If this backup is successful, ON-Bar returns 178.

To salvage the logical logs, use the **onbar -b -s -O** command.

Warning: *If you back up logical logs that contain changes to a blobspace while it is offline, the simple large objects in that blobspace will not be restorable. If an offline blobspace has not changed, it will be restorable.*

XPS

Assigning a Name to a Backup Session

To assign a name to a backup session, use the following command:

```
onbar -b -q session1
```

To monitor the backup session, use the **onstat -g bus** command.

XPS

Performing a Physical Backup

Use the **-p** option to perform either a level-0 or incremental backup of storage spaces only without also backing up the logical logs, as the following command shows. In the backup command, you can list the storage spaces on the command line or in a file.

```
onbar -b -p
```

To restore data, you must have previously backed up the logical logs. For more information, see [“Backing Up Logical Logs on Extended Parallel Server” on page 4-25](#).

XPS

Backing Up a Dbslice

You can specify storage spaces individually or with a *dbslice* name. Specifying storage spaces with a *dbslice* name simplifies backup commands. The *dbslice* name is translated to the names of its component storage spaces. If you back up a *dbslice*, ON-Bar backs up all the storage spaces in that *dbslice*.

To back up all dbspaces in a *dbslice* named **fin_slice**, use the following command:

```
onbar -b fin_slice
```

Backing Up Table Types

Figure 4-2 discusses backup scenarios for six table types. For more information about the table types, see the chapter on data storage in the *Administrator's Guide*.

Figure 4-2
Backing Up Table Types

Table Type	IDS	XPS	Can You Back Up This Type of Table?
Standard	✓	✓	Yes.
Temp	✓	✓	No.
Scratch		✓	No.
Operational		✓	Yes. If you use pload express mode to load an operational table, you cannot reliably restore the data unless you have done a level-0 backup after the load. It is not enough to just back up the logical logs.
Raw	✓	✓	Yes. If you update a raw table, you cannot reliably restore the data unless you perform a level-0 backup after the update. Backing up only the logical logs is not enough.
Static		✓	Yes.



Important: Perform a level-0 backup before you alter a raw, static, or operational table to type STANDARD.

Backing Up Logical Logs

For background information, see “What Is a Logical-Log Backup?” on [page 1-5](#). You can either back up the logical logs separately or with storage spaces. It is recommended that you back up the logical logs as soon as they fill so that you can reuse them and to protect against data loss if the disks that contain the logs are lost. If the log files fill, the database server pauses until you back up the logical logs.

You can either back up the logical logs manually or start a continuous logical-log backup. Logical-log backups are always level 0. After you close the current logical log, you can back it up.

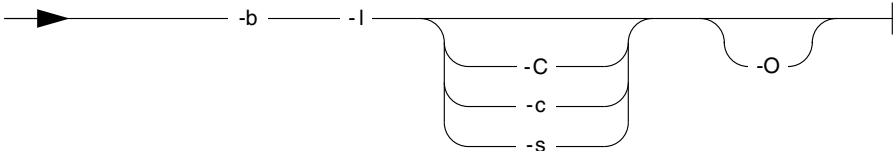
Backing Up Logical Logs on Dynamic Server

If you do not use whole-system backups, you must back up logical logs because you must restore both the storage spaces and logical logs.

If you perform whole-system backups and restores, you can avoid restoring logical logs. It is recommended that you also back up the logical logs when you use whole-system backups. These log backups allow you to recover your data to a time after the whole-system backup, minimizing data loss. The following diagram shows the syntax for **onbar -b -l** commands.

[Backing Up Logical Logs](#)

[Back to Syntax of ON-Bar Commands](#)
p. [4-4](#)



Command	Purpose	Key Considerations
-b -l	Performs a backup of full logical-log files	The current logical-log file is not backed up. If you are using ISM, it also backs up the ISM catalog.
-b -l -c	Closes and backs up the current logical log as well as the other full logical logs	None.

(1 of 2)

Command	Purpose	Key Considerations
-b -l -C	Starts a continuous log backup	Reserve a dedicated storage device and terminal window because the continuous log backups run indefinitely waiting for logical logs to fill. To stop a continuous log backup, kill the ON-Bar process with an interrupt (^C or SIGTERM).
-b -l -O	Overrides normal logical backup restrictions such as when a blobspace is offline	If a log backup occurs when blobspaces are offline, return code 178 displays in the ON-Bar activity log.
-b -l -s	Salvages any logical logs that are still on disk after a database server failure	If possible, use this option before you replace a damaged disk. If you use onbar -r to perform a cold restore on an undamaged disk, ON-Bar automatically salvages the logical logs. For information, see “Salvaging Logical Logs” on page 6-19 .

(2 of 2)

Performing a Continuous Backup of Logical Logs

You can start a continuous logical-log backup in the following ways:

- Specify **onbar -b -l -C**.
- Set the ALARMPROGRAM parameter to the full path for **log_full.sh** on UNIX or **log_full.bat** on Windows.
- Set the ALARMPROGRAM parameter to the full path for **alarmprogram.sh** on UNIX or **alarmprogram.bat** on Windows and set the BACKUPLGDS parameter within the file to **y**.
- You can write your own event alarm and set ALARMPROGRAM to it. For more information, see [“ALARMPROGRAM” on page 9-7](#) and the *Administrator’s Reference*.

After the continuous logical-log backup starts, it runs indefinitely waiting for logical logs to fill. To stop the continuous logical-log backup, kill the ON-Bar process.

If an error occurs while the continuous logical-log backup is running, it stops. If it stops, reissue the **onbar -b -l -C** command.

Tip: Reserve a dedicated storage device to improve performance during continuous logical-log backups.



Performing a Manual Backup of Logical Logs

To start a manual logical-log backup, use the **onbar -b -l** command. If you set ALARMPROGRAM to **no_log.sh** or **no_log.bat**, you must initiate a logical-log backup manually.

To back up the current logical-log file, use the **onbar -b -l -c** command.

Using ALARMPROGRAM to Set the Log Backup Mode

Use the ALARMPROGRAM configuration parameter to control continuous log backups. If ALARMPROGRAM is set to **log_full.sh** or **log_full.bat**, when a logical-log file fills, the database server triggers event alarm 23. This event alarm calls **onbar -b -l** to back up the full logical-log file. Restart the database server after you change the value of ALARMPROGRAM.

If you do not set ALARMPROGRAM, or if you set it to **\$INFORMIXDIR/etc/log_full.sh**, ON-Bar performs continuous log backups. To turn off continuous log backups, set ALARMPROGRAM to **\$INFORMIXDIR/etc/no_log.sh**. ♦

To use ALARMPROGRAM to start continuous log backups, set it to **%INFORMIXDIR%\etc\log_full.bat**. To turn off continuous log backups, set ALARMPROGRAM to **%INFORMIXDIR%\etc\no_log.bat**. ♦

To generate email or pager messages to a designated DBA when a specific error level is triggered, set ALARMPROGRAM to **\$INFORMIXDIR/etc/alarmprogram.sh** or **%INFORMIXDIR%\etc\alarmprogram.bat**. Then edit the file to turn automatic logging on, set the level of errors, and insert the e-mail address of the DBA.

UNIX

Windows

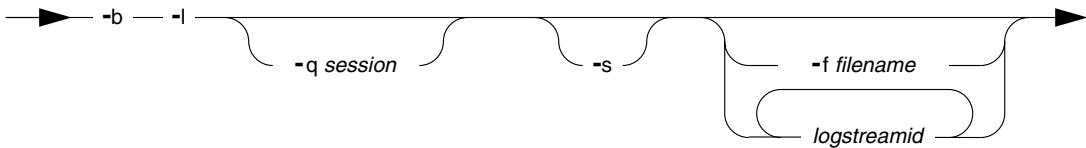
Backing Up Logical Logs on Extended Parallel Server

You must back up logical logs even when you use nonlogging tables because you must restore both storage spaces and logical logs.

Use **onstat -g bus** or **onstat -g bus_sm** to monitor logical logs and **onbar-worker** processes on each coserver in the current backup session. For more information, see “Monitoring the Backup Scheduler Status” on page 8-17.

Backing Up Logical Logs

Back to Syntax of ON-Bar Commands
p. 4-4



Command	Purpose	Key Considerations
-b -l	Performs a backup of full logical-log files	The current logical-log file is not backed up. If you are using ISM, it also backs up the ISM catalog.
-b -l -f filename	Backs up the logstreamids that are listed in the text file whose pathname <i>filename</i> provides Use this option to avoid entering a long list of <i>logstreamids</i> every time that you use this option.	The filename can be any valid UNIX or Windows filename, including simple (<i>listfile_1</i>), relative (<i>../backup_lists/listfile_2</i> or <i>..\backup_lists\listfile</i>), and absolute (<i>/usr/informix/backup_lists/listfile3</i> or <i>c:\informix\backup_lists\listfile3</i>) filenames. The file can list multiple logstreamids per line.

(1 of 2)

Command	Purpose	Key Considerations
-b -l <i>logstreamid</i>	Uniquely identifies a logical-log stream that a given XPS coserver generates	If you supply more than one <i>logstreamid</i> , separate each item in the list with a space. A logstreamid is the same as a coserver ID.
-b -l -q <i>session</i>	Allows you to assign a name to the log backup session This name appears in the onstat utility so that you can follow the progress of the log backup.	<DBSERVERNAME><random_number> is the default session name. The session name must be unique and can be up to 128 characters.
-b -l -s	Salvages any logical logs that are still on disk after a database server failure	If possible, use this option before you replace a damaged disk. If you use onbar -r to perform a cold restore on an undamaged disk, ON-Bar automatically salvages the logical logs. For more information, see “Salvaging Logical Logs” on page 6-19 .

(2 of 2)

Performing a Manual Backup of Logical Logs

Use the LOG_BACKUP_MODE configuration parameter to specify whether to back up full logical-log files automatically or manually. If you change the value of LOG_BACKUP_MODE, you must restart the database server before the change takes effect.

If you set LOG_BACKUP_MODE to MANUAL, you must initiate a logical-log backup manually. To back up filled logical-log files manually, use the following command:

```
onbar -b -l
```

If you want to assign a session name to the log backup, use the **-q** option, as follows:

```
onbar -b -l -q "my_logbackup"
```

To back up the current log, use the following commands:

```
xctl onmode -l  
onbar -b -l
```


Starting Continuous Logical-Log Backups

On Extended Parallel Server, you can start a continuous logical-log backup by setting the LOG_BACKUP_MODE configuration parameter to CONT. Whenever a logical-log file fills, the Backup Scheduler automatically starts an **onbar-worker** process, if one is not already active, and assigns the log backup to it.

To stop a continuous logical-log backup, you must suspend the backup session. For example, the following command turns off continuous logical-log backup for the session “**log backup 1**” on coserver 1. In a multicoserver environment, you might need to turn off logical-log backup on each coserver.

```
onbar off -q "log backup 1"
```

To find the logical-log backup session, use the **onstat -g bus** command. For more information, see [“Starting and Stopping ON-Bar Sessions” on page 8-14](#).

Preventing Logical-Log Backups in a Test System

If you set LOG_BACKUP_MODE to NONE, you cannot back up or restore logical logs, and log salvage does not work. Although you can continue to back up storage spaces, you cannot restore them. The only reason to set LOG_BACKUP_MODE to NONE is to test your system. Do not use LOG_BACKUP_MODE = NONE in a production system.

Monitoring Logical-Log Backups

To find out if a logical-log file is ready to be backed up, check the flags field of **onstat -l**. After the logical-log file is marked as backed up, it can be reused. When the flags field displays any of the following values, the logical-log file is ready to be backed up:

```
U-----
U-----L
```

The value **U** means that the logical-log file is used. The value **L** means that the last checkpoint occurred when the indicated logical-log file was current. The value **C** indicates the current log. If **B** appears in the third column, the logical-log file is already backed up and can be reused.

```
U-B---L
```

The following example shows the output of **onstat -l** when you use it to monitor logical logs on the database server:

```

Logical Logging
Buffer bufused  bufsize  numrecs  numpages  numwrits  recs/pages  pages/io
L-2  0          16        1          1          1          1.0        1.0
Subsystem      numrecs  Log Space used
OLDRSAM        1        32
address number  flags    uniqid   begin    size    used    %used
a038e78  1      U-B----  1       10035f   500     500    100.00
a038e94  2      U-B----  2       100553   500     500    100.00
a038eb0  3      U---C-L  3       100747   500     366    73.20
a038ecc  4      F-----  0       10093b   500     0       0.00
a038ee8  5      F-----  0       100b2f   500     0       0.00
a038f04  6      F-----  0       100d23   500     0       0.00

```

For information about the **onstat -l** fields, see [“onstat -l” on page B-6](#).

Warning: If you turn off continuous logical-log backup, you must monitor your logical logs carefully and start logical-log backups as needed.

The flag values U---C-L or U---C-- represent the current logical log. While you are allowed to back up the current logical log, doing so forces a log switch that wastes logical-log space. Wait until a logical-log file fills before you back it up. ♦

On Extended Parallel Server, use the **xctl onstat -l** utility to monitor logical logs on all coservers. ♦

Salvaging Logical-Log Files

Use **onbar -b -l -s** to salvage the logs.

ON-Bar salvages logical logs automatically before a cold restore unless you specify a physical restore only. ON-Bar salvages the logical logs that are used before it restores the root dbspace. To make sure that no data is lost before you start the cold restore, manually salvage the logical logs in the following situations:

- If you must replace the media that contains the logical logs
If the media that contains the logical logs is no longer available, the log salvage will fail, resulting in data loss.
- If you plan to specify a physical restore only (**onbar -r -p**)

For more information, see [“Salvaging Logical Logs” on page 6-19](#) and [“Performing a Cold Restore” on page 6-19](#).



IDS

XPS

XPS

Skipping Logical Replay

The database server automatically skips logical replay when it is not necessary during a warm restore. For example, if you lose a dbspace that contains a large table that has not changed since its last backup, you can quickly restore it without replaying the logical logs. If *all* dbspaces being restored on a coserver meet the following criteria, logical replay is skipped for the warm restore:

- A backup or set of incremental backups exists for the dbspaces.
- No logging activity has occurred for the dbspaces since the last backup.

If dbspaces are being restored on multiple coservers, logical replay is skipped on those coservers where no dbspaces need it and performed on those coservers where at least one dbspace needs it.

In the following **onstat -d** example, the **S** flag shows that dbspace **dbnoplay** is a candidate for skipping logical replay. This **S** flag disappears the first time that a logging operation occurs in that dbspace.

```
Dbspaces
address number flags fchunk nchunks flags owner name
a66c140 1 1 1 1 N informix rootdbs
a68bea8 2 20001 2 1 N S informix dbnoplay
```

The database server always replays the logical logs during a cold restore.

Restoring Nonlogging Databases and Tables

Warning: If you do not use logging for your databases or tables, ON-Bar can only restore the data up to the time it was most recently backed up. Changes made to data since the last standard backup are not restorable. If you do not use logging, you would need to redo lost transactions by hand.

If logical-log backups are disabled because LTAPEDEV is set to **/dev/null** or **NUL**, you can restore only whole-system backups. ♦

If logical-log backups are disabled because LOG_BACKUP_MODE is set to **NONE**, restores are not possible. ♦

Warning: It is strongly recommended that you do not set LTAPEDEV to **/dev/null** or **NUL**, or LOG_BACKUP_MODE to **NONE**.



IDS

XPS



Understanding ON-Bar Backup Processes

This section explains how ON-Bar performs backup operations on the database server. You can perform a backup when the database server is in online or quiescent mode. The original ON-Bar process is called the *driver*, and each new ON-Bar process that it creates is called an **onbar_d** *child* process.

Backup Sequence on Dynamic Server

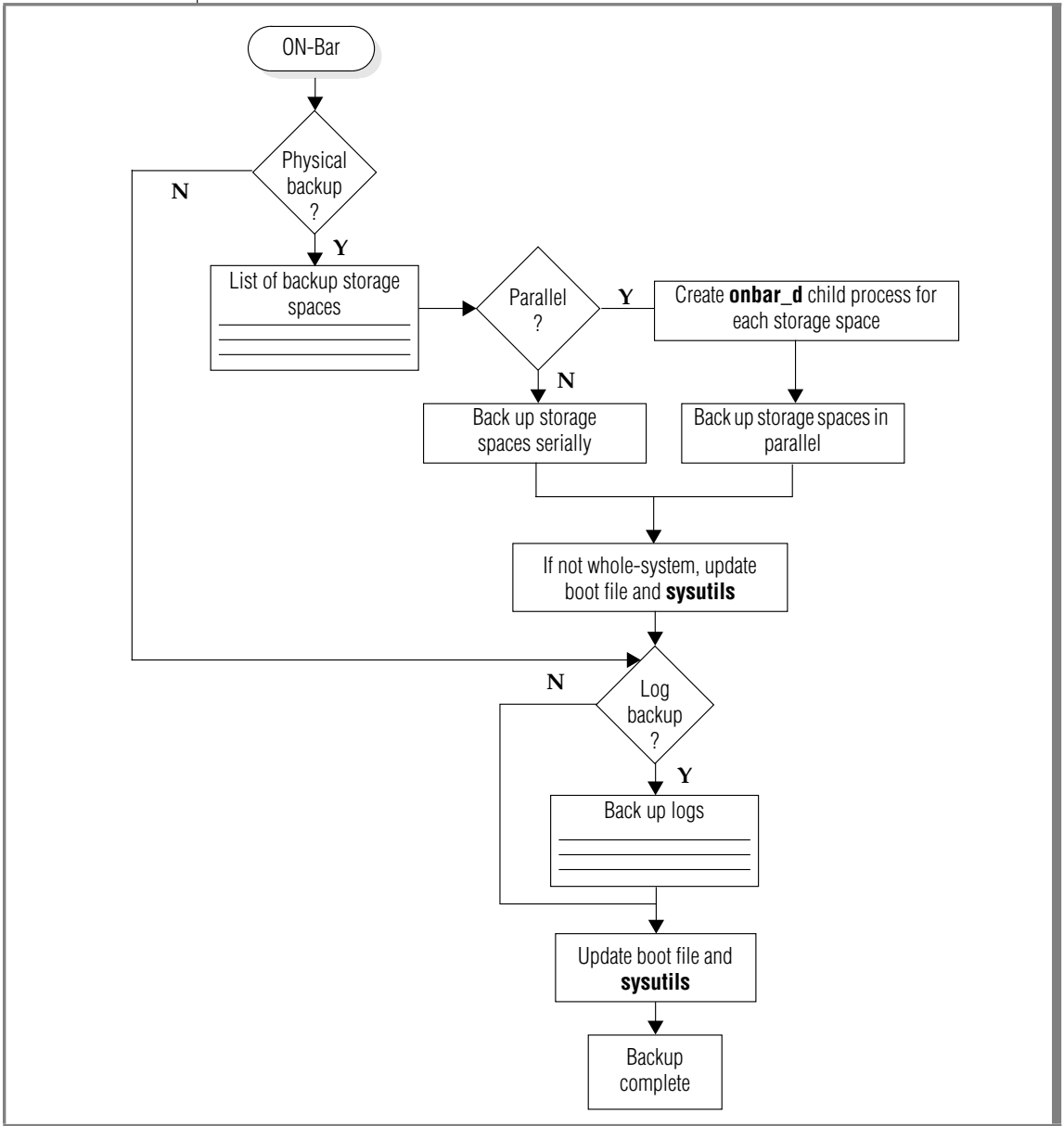
[Figure 4-3 on page 4-31](#) describes the ON-Bar backup sequence. When you issue a backup command, the **onbar-driver** builds a list of storage spaces and creates a backup session.

In a parallel backup (if `BAR_MAX_BACKUP` is not set to 1), the **onbar-driver** starts one or more **onbar_d** child processes and assigns backup tasks to them. Each **onbar_d** child process backs up one storage space. Each **onbar_d** child disappears when the backup of its storage space is done. The **onbar-driver** keeps creating new children until all the storage spaces are backed up. Then the **onbar-driver** backs up the logical logs.

If you specify a whole-system backup or set `BAR_MAX_BACKUP` to 1, the **onbar_driver** backs up the storage spaces and logical logs serially. No **onbar_d** child processes are created.

When the backup is complete, the **onbar-driver** determines whether an error occurred and returns a status in the ON-Bar activity log. After each object is backed up, information about it is added to the emergency boot file on the database server and to the **sysutils** database.

Figure 4-3
ON-Bar Backup Process on Dynamic Server



Backup Sequence on Extended Parallel Server

[Figure 4-4 on page 4-33](#) describes the ON-Bar backup sequence. The **onbar-driver** builds and sends a list of storage spaces to the Backup Scheduler. The Backup Scheduler creates a backup session, might start one or more **onbar-worker** processes, and assigns backup tasks to the **onbar-worker** processes.

Once all the storage spaces are backed up, the **onbar-driver** sends a list of logstreams (logical-log data) to the Backup Scheduler that assigns the tasks to **onbar-worker** processes. Each **onbar-worker** process is associated with a coserver and a storage-manager instance. Once an **onbar-worker** process starts, it might be active after the backup or restore session is completed. An **onbar-worker** can process parts of several backup or restore sessions in its lifetime.

Each **onbar-worker** transfers data between the database server and the storage manager until the backup request is fulfilled. When an **onbar-worker** completes its task, it waits for the next task from the Backup Scheduler. If no new task is assigned in a user-specified amount of time, the **onbar-worker** shuts down. You can set the number of minutes that the **onbar-worker** processes wait in `BAR_IDLE_TIMEOUT` in the `ONCONFIG` file.

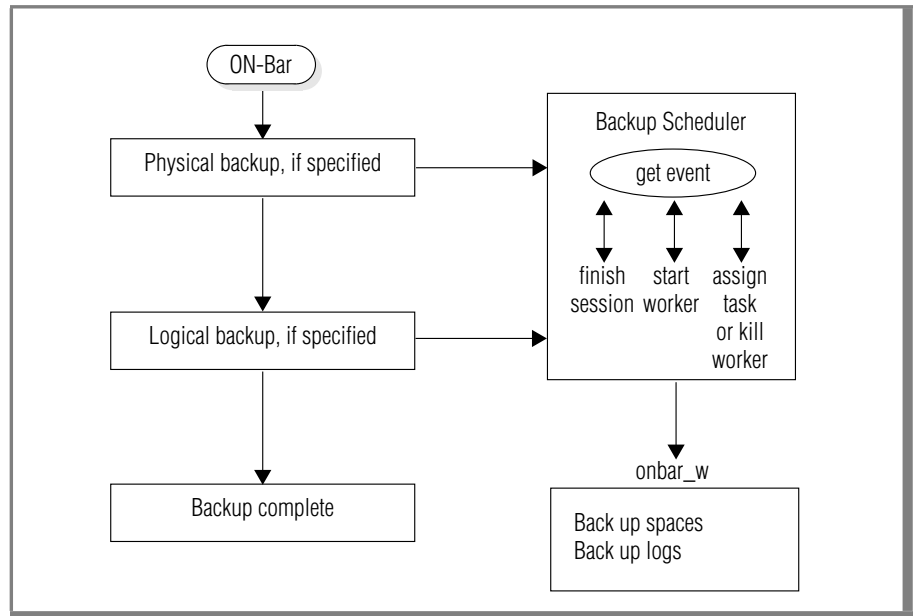
If the Backup Scheduler has new tasks to assign and not enough **onbar-worker** processes are running to complete the task, it calls the **start_worker** script to start one or more new **onbar-worker** processes.

It is recommended that you start **onbar-worker** processes automatically, but if you want to start them manually, see [“Using start_worker.sh to Start onbar_worker Processes Manually” on page 8-7](#). If you have set `BAR_WORKER_MAX = 0`, you must start a new **onbar-worker** manually.

After each object is backed up, ON-Bar updates the emergency backup boot file on the coserver that is local to the **onbar-worker** and the **sysutils** database. The emergency backup boot file is on the coserver of the **onbar-worker** that backed it up.

To monitor the status of backups, restores, and **onbar-worker** activities, use the **onstat -g bus** or **onstat -g bus_sm** options or check the Backup Scheduler tables in the **sysmaster** database. For more information, see [“Monitoring the Backup Scheduler Status”](#) on page 8-17 and [“Backup Scheduler SMI Tables”](#) on page 10-11.

Figure 4-4
ON-Bar Backup Process on Extended Parallel Server



Verifying Backups

In This Chapter	5-3
Verifying Backups with archecker.	5-3
Using archecker to Verify Backups	5-4
Preparing to Verify Backups	5-6
Syntax	5-7
Estimating the Amount of Temporary Space for archecker	5-8
Verification Examples	5-10
Performing Verification Only.	5-10
Performing a Point-in-Time Verification	5-10
Verifying a Whole-System Backup	5-10
Verifying Blobspaces	5-10
Verifying Sbspaces	5-10
Interpreting Verification Messages	5-11
Sample Verification Message in the ON-Bar Activity Log	5-11
Sample Verification Message in the archecker Message Log	5-11
What To Do If Backup Verification Fails	5-12
Backups with Corrupt Pages	5-12
Backups with Corrupt Control Information.	5-12
Backups with Missing Data	5-12
Backups of Inconsistent Database Server Data	5-12
Procedures for Fixing Backup Verification Problems	5-13

In This Chapter

This chapter describes the **archecker** utility for checking the validity and completeness of backups. To ensure that you can restore a backup safely, issue the **onbar -v** command, which calls the **archecker** utility.

Verifying Backups with archecker

You access the **archecker** utility when you use the **onbar -v** command. You can use **archecker** with the database server in any mode. The **archecker** utility verifies that all pages required to restore a backup exist on the media in the correct form. After you successfully verify a backup, you can restore it safely. If **archecker** shows problems with the backup, contact Technical Support.

The **archecker** utility verifies standard and whole-system backups. The **archecker** utility cannot verify logical-log backups. The **archecker** utility does *not* perform a restore.

Figure 5-1 shows how ON-Bar and **archecker** verify a backup. The **archecker** utility verifies level-0 backups on all database servers. The following steps correspond to the circled numbers in Figure 5-1.

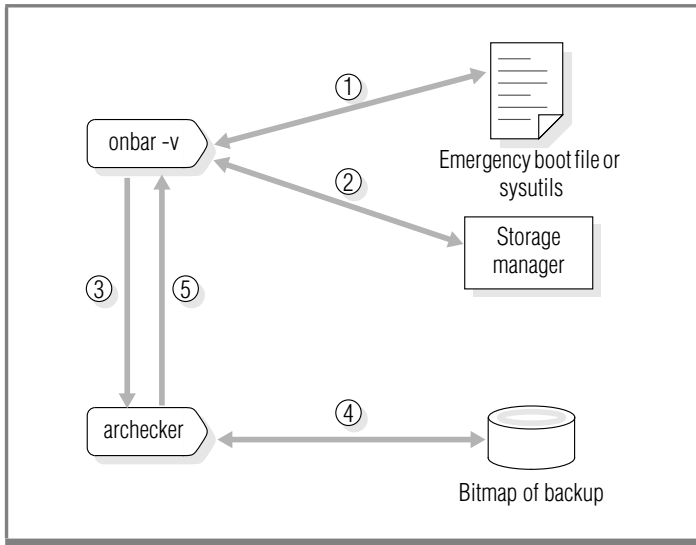


Figure 5-1
How ON-Bar Verifies a Backup

Using archecker to Verify Backups

When the user issues an **onbar -v** command, the following sequence of actions occurs:

1. ON-Bar uses the emergency boot file if the database server is offline or in microkernel mode or the **sysutils** database if the database server is online or quiescent to determine which backup to verify.
2. ON-Bar requests and retrieves the backup data from the storage manager.
3. ON-Bar forwards the backup data to **archecker**.

4. The **archecker** utility scans the backup data and creates a bitmap of the pages. During the scan phase, **archecker** verifies the following types of problems:
 - Backups with corrupt pages
 - Backups with corrupt control information
 - Backups with missing pages that have been added since the last level-0 backup
 - Retrieval of the wrong backup objects

An example of retrieving the wrong backup object is if ON-Bar requests the **rootdbs** backup from last Wednesday but the storage manager retrieves the **rootdbs** backup from last Tuesday.
5. After it completes the scan, **archecker** uses this bitmap to verify the backup and records the status in the **archecker** message log. ON-Bar also records this status in the ON-Bar activity log.
6. When a backup is verified, ON-Bar inserts a row into the emergency boot file with the backup copy ID and the verification date, and updates the **ins_verify** and **ins_verify_date** rows of the **bar_instance** table in the **sysutils** database. For more information, see [“The bar_instance Table” on page 10-5](#).

During the verification phase, **archecker** verifies that all the pages for each table are present and checks the partition pages, the reserved pages, the chunk-free list, blobspaces, sbspaces, and extents. The **archecker** utility also checks the free and used counts, verifies that the page stamps match and that no overlap exists in the extents.

Archecker writes temporary files in the directory that the AC_STORAGE parameter specifies. For information, see [“AC_STORAGE” on page 9-5](#).

Preparing to Verify Backups

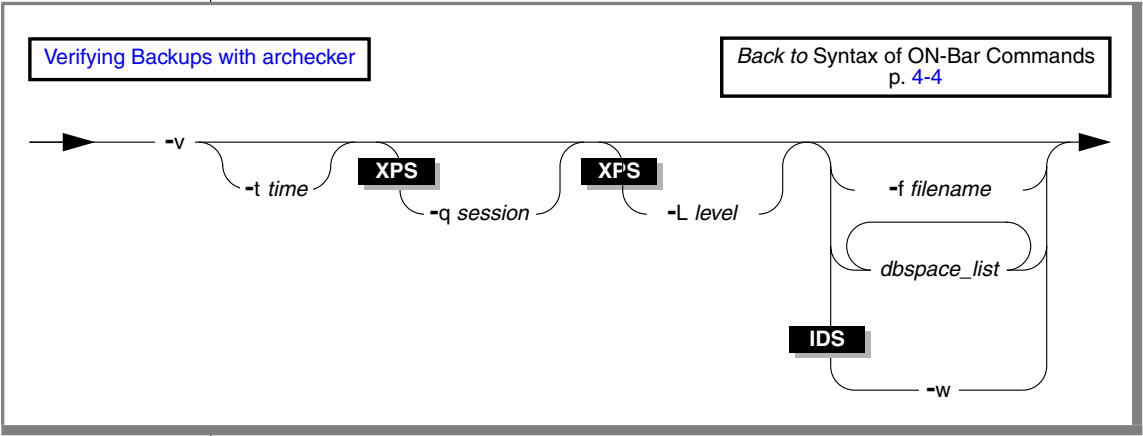
This section describes preparing to verify backups.

To prepare to verify backups

1. Review the **ac_config.std** file that contains default **archecker** configuration parameters to see if you want to change the location of the directories for the **archecker** message log and temporary files.
2. Set the **AC_CONFIG** environment variable to specify the path for the **archecker** configuration file (either **ac_config.std** or user defined).
For more information on setting the path, see [“Setting archecker Configuration Parameters in AC_CONFIG” on page 9-4](#).
3. Estimate the amount of temporary space needed for storing the **archecker** temporary files.
For more information, see [“Estimating the Amount of Temporary Space for archecker” on page 5-8](#).
4. Use the **onbar -v** option to verify that an existing backup will restore properly.
For examples, see [“Verification Examples” on page 5-10](#).
5. To see whether **archecker** verified the backup successfully, check the **archecker** return code in the ON-Bar activity log and the messages in the **archecker** message log.

Syntax

This diagram shows the **onbar -v** syntax.



Element	Purpose	Key Considerations
-v	Verifies a backup If verification is successful, you can restore the storage spaces safely.	Specify onbar -v to verify the backup. You can perform a point-in-time verification. You cannot verify the logical logs. You must specify the -b or -v parameter first. You can verify a whole-system or physical-only backup (IDS).
dbspace_list	Names a list of storage spaces to be backed up or verified	If you enter more than one storage-space name, use a space to separate the names. On XPS, if you enter a dbslice name, it verifies all the dbspaces in that dbslice.
-f filename	Verifies the storage spaces that are listed in the text file whose pathname <i>filename</i> provides Use this option to avoid entering a long list of storage spaces every time that you verify them.	You can use any valid UNIX or Windows pathname and filename. For the format of this file, see Figure 4-1 on page 4-17 . The file can list multiple storage spaces per line.

Element	Purpose	Key Considerations
-L level	Specifies the level of backup to verify (XPS): <ul style="list-style-type: none">■ 0 for a complete backup■ 1 for changes since the last level-0 backup■ 2 for changes since the last level-1 backup	The archecker utility fully verifies level-0 backups on all database servers and performs limited verification of level-1 and level-2 backups.
-q session	Assigns a name to the verify session (XPS)	<DBSERVERNAME><random_number> is the default session name. The session name must be unique and can be up to 126 characters. This name appears in the onstat utility so that you can follow the progress of the verify session.
-t time	Specifies the date and time to which dbspaces are verified	How you enter the time depends on your current GLS locale convention. If the GL_DATETIME environment variable is set, you must specify the date and time according to that variable. If the GLS locale is not set, use ANSI-style date format: YYYY-MM-DD HH:MM:SS .
-w	Verifies a whole-system backup	Available on IDS only.

(2 of 2)

Estimating the Amount of Temporary Space for archecker

The **archecker** utility requires about 15 megabytes of temporary space for a medium-size system (40-50 gigabytes) and 25 megabytes for a large system. This temporary space is stored on the file system in the directory that the AC_STORAGE parameter specifies, not in the dbspaces. The temporary files contain bitmap information about the backup and copies of partition pages, chunk-free pages, reserved pages, and optionally, blob-free map pages and debugging information. The **archecker** utility must have permissions to the temporary directory.

If the backup is verified successfully, these files are deleted. If the backup fails verification, these files remain. Copy them to another location so that Technical Support can review them.

If your database server contains only dbspaces, use the following formula to estimate the amount of temporary space in kilobytes for the **archecker** temporary files:

$$\text{space} = (130 \text{ KB} * \text{number_of_chunks}) + (\text{pagesize} * \text{number_of_tables}) + (.05 \text{ KB} * \text{number_of_logs})$$

If your database server contains blobspaces or sbspaces, use the following formula to estimate the amount of temporary space for the **archecker** temporary files:

$$\text{space} = (130 \text{ KB} * \text{number_of_chunks}) + (\text{pagesize} * \text{number_of_tables}) + (.05 \text{ KB} * \text{number_of_logs}) + (\text{pagesize} * (\text{num_of_blobpages}/252))$$



<i>number_of_chunks</i>	is the maximum number of chunks that you estimate for the database server.
<i>pagesize</i>	is the system page size in kilobytes.
<i>number_of_tables</i>	is the maximum number of tables that you estimate for the database server.
<i>number_of_logs</i>	is the number of logical logs on the database server.
<i>num_of_blobpages</i>	is the number of blobpages in the blobspaces or the number of sbspaces. (If your database server contains sbspaces, substitute <i>num_of_blobpages</i> with the number of sbspaces.) (IDS only)

For example, you would need 12.9 megabytes of temporary disk space on a 50-gigabyte system with a page size of 2 kilobytes. This system does not contain any blobspaces, as the following statement shows:

$$13,252 \text{ KB} = (130 \text{ KB} * 25 \text{ chunks}) + (2 \text{ KB} * 5000 \text{ tables}) + (.05 \text{ KB} * 50 \text{ logs}) + (2 \text{ KB} * 0)$$

To convert kilobytes to megabytes, divide the result by 1024:

$$12.9 \text{ MB} = 13,252/1024$$

Verification Examples

The following examples show how to verify an existing backup and how to verify immediately after backing up.

Performing Verification Only

To verify a backup of all storage spaces, use the **onbar -v** command. The logical logs are not verified.

To verify the backed-up storage spaces listed in the file **bkup1**, use the following command:

```
onbar -v -f /usr/backups/bkup1
```

Performing a Point-in-Time Verification

To perform a point-in-time verification of a backup, use the following command with the *datetime* value in quotes:

```
onbar -v -t "2001-12-10 10:20:50"
```

IDS

Verifying a Whole-System Backup

To verify a whole-system backup, use the following command:

```
onbar -v -w
```

IDS

Verifying Blobspaces

The **onbar -v** command cannot verify the links between data rows and simple large objects in a blob space. Use the **oncheck -cD** command instead to verify the links in a blob space. For information on **oncheck**, see the *Administrator's Reference*.

IDS

Verifying Sbspaces

The **onbar -v** command verifies only the smart-large-object extents in an sbspace. For a complete check, use the **oncheck -cS** command. For information on **oncheck**, see the *Administrator's Reference*.

Interpreting Verification Messages

When you verify a backup, ON-Bar writes summary messages to the **bar_act.log** that report which storage spaces were verified and whether the verification succeeded or failed. The **archecker** utility writes detailed messages to the **ac_msg.log**. Technical Support uses the **ac_msg.log** to diagnose problems with backups and restores.

Sample Verification Message in the ON-Bar Activity Log

The level-0 backup of dbspace **db2.2** passed verification, as follows:

```
Begin backup verification of level0 for db2.2 (Storage Manager Copy ID:##)
Completed level-0 backup verification successfully.
```

The level-0 backup of **rootdbs** failed verification, as follows:

```
Begin backup verification of level0 for rootdbs (Storage Manager Copy
ID:##) .
ERROR: Unable to close the physical check: error_message.
```

Sample Verification Message in the archecker Message Log

More detailed information is available in the **archecker** message log, as follows:

```
STATUS: Scan PASSED
STATUS: Control page checks PASSED
STATUS: Starting checks of dbspace db2.2.
STATUS: Checking db2.2:TBLSpace
.
.
STATUS: Tables/Fragments Validated: 1
Archive Validation Passed
```

What To Do If Backup Verification Fails

If a backup fails verification, do not attempt to restore it. The results are unpredictable and range from corruption of the database server to a failed restore because ON-Bar cannot find the backup object on the storage manager. In fact, the restore might appear to be successful but it hides the real problem with the data or media.

The three different types of corrupt backups are as follows:

- Backups with corrupt pages
- Backups with corrupt control information
- Backups with missing data

Backups with Corrupt Pages

If the pages are corrupt, the problem is with the databases rather than with the backup or the media. Run **oncheck -cd** on any tables that produce errors and then redo the backup and validation. To check extents and reserved pages, run **oncheck -ce** and **oncheck -cr**.

Backups with Corrupt Control Information

In this case, all the data is correct, but some of the backup control information is incorrect, which could cause problems with the restore. Ask Technical Support for assistance.

Backups with Missing Data

When a backup is missing data, it might not be recoverable. After a data loss, try to restore from an older backup. Then restore the current logical logs.

Backups of Inconsistent Database Server Data

There are cases where **archecker** returns “success” to ON-Bar but shows “failure” in the **archecker** message logs. This situation occurs when **archecker** verifies that ON-Bar backed up the data correctly, but the database server data was invalid or inconsistent when it was backed up.

Procedures for Fixing Backup Verification Problems

Follow these steps when a backup fails verification. The first procedure diagnoses why a backup failed verification; the second procedure verifies an expired backup; and the third procedure verifies a backup with missing data.

To diagnose why a backup failed verification

1. Verify that the **AC_CONFIG** environment variable and the contents of the **archecker** configuration file are set correctly. If these variables are set incorrectly, the ON-Bar activity log displays a message.
2. Immediately redo the backup onto different media.
Do *not* reuse the original backup media because it might be bad.
3. Do not use any backups based on this backup. If the level-0 backup is bad, do not use the corresponding level-1 and level-2 backups.
4. Verify this new backup. If verification succeeds, you will be able to restore the storage spaces with confidence.
5. Use your storage manager to expire the backup that failed verification and then run the **onsmsync** utility.

For more information on expiring data from the storage manager, see your storage-manager documentation or the *IBM Informix Storage Manager Administrator's Guide*. For more information, see [“Using the onsmsync Utility” on page 8-10](#).

6. If verification fails again, call Technical Support and provide them with the following information:
 - Your backup tool name (ON-Bar)
 - The database server **online.log**
 - The archecker message log
 - The **AC_STORAGE** directory that contains the bitmap of the backup and copies of important backed-up pages

If only part of the backup is corrupt, Technical Support can help you determine which portion of the backup can be restored in an emergency.

7. Technical Support might advise you to run **oncheck** options against a set of tables. (See [“Backups with Corrupt Pages” on page 5-12](#).)

To verify an expired backup

1. Check the status of the backup save set on the storage manager. If the storage manager has expired the backup save set, the **archecker** utility cannot verify it.
2. Use the storage-manager commands for activating the expired backup save set. See your storage-manager documentation or the *IBM Informix Storage Manager Administrator's Guide*.
3. Retry the backup verification: **onbar -v**.

To restore when a backup is missing data

1. Choose the date and time of an older backup than the one that just failed. To perform a point-in-time verification, use the following command:

```
onbar -v -t datetime dbspace1
```

2. If the older backup passes verification, perform a point-in-time physical restore using the same datetime value, then perform a log restore, as follows:

```
onbar -r -p -t datetime dbspace1  
onbar -r -l
```

3. To prevent the storage manager from restoring a backup that failed verification, use your storage manager to expire the bad backup and then run the **onsmsync** utility. The **onsmsync** utility removes the bad backup from the emergency boot file and the **sysutils** database.

Restoring Data with ON-Bar

In This Chapter	6-3
What Types of Restores Does ON-Bar Perform	6-3
What Is a Warm Restore?	6-3
What Is a Cold Restore?	6-4
What Is a Mixed Restore?	6-5
What Is a Parallel Restore?	6-5
What Is a Whole-System Restore?	6-5
What Is a Point-in-Time Restore?.	6-6
What Is an Imported Restore?.	6-6
What is a Rename Chunks Restore?.	6-6
What Is a Restartable Restore?	6-7
Using the Pre-Recovery Checklist	6-7
Monitoring Restores	6-8
Ensuring That Storage Devices Are Available	6-9
Restoring Save Sets with ISM	6-10
Performing a Complete Restore	6-10
Performing a Physical-Only or Logical-Only Restore	6-13
Using IBM Informix Server Administrator to Restore Data	6-15
Examples of ON-Bar Restore Commands	6-16
Performing a Restore	6-16
Restoring Specified Storage Spaces.	6-16
Performing a Logical Restore.	6-17
Restoring Data on Extended Parallel Server	6-17
Performing a Physical Restore Followed By a Logical Restore	6-18
Salvaging Logical Logs.	6-19
Performing a Cold Restore	6-19
Performing a Whole-System Restore	6-21
Restoring Data to a Point in Time	6-22

Restoring from an Older Backup	6-22
Performing a Point-in-Log Restore	6-23
Restoring Online Storage Spaces	6-23
Re-Creating Chunk Files During a Restore	6-24
Restoring a Dropped Storage Space	6-25
Restoring Data when Reinitializing the Database Server	6-26
Renaming Chunks During a Restore	6-27
Key Considerations	6-27
New-Chunk Requirements	6-28
Syntax	6-28
Examples of Renaming Chunks During a Restore	6-30
Renaming Chunks with Command Line Options	6-30
Renaming Chunks with a File	6-30
Renaming Chunks While Specifying Other Options	6-31
Renaming a Chunk to a Nonexistent Device	6-31
Transferring Data with the Imported Restore	6-32
Importing a Restore	6-34
Initializing High-Availability Data Replication with ON-Bar	6-36
Restoring Table Types	6-38
Using Restartable Restore to Recover Data	6-39
Restartable Restore Example	6-40
Restarting a Restore	6-41
Interaction Between Restartable Restore and BAR_RETRY Value	6-42
Restarting a Logical Restore	6-43
Resolving a Failed Restore	6-44
Understanding ON-Bar Restore Processes	6-46
Warm-Restore Sequence on Dynamic Server	6-46
Cold-Restore Sequence on Dynamic Server	6-48
Warm-Restore Sequence on Extended Parallel Server	6-50
Cold-Restore Sequence on Extended Parallel Server	6-52

In This Chapter

This chapter describes the different types of restores that ON-Bar performs.

What Types of Restores Does ON-Bar Perform

You can restore storage spaces stored in both raw and cooked files. If your system contains primary and mirrored storage spaces, ON-Bar writes to both the primary and mirrored chunks at the same time during the restore. (Mirroring is a strategy that pairs a *primary* chunk of one storage space with an equal-sized *mirrored chunk*.)



Important: *You cannot specify temporary dbspaces in a warm or cold restore. When you restore the critical dbspaces (for example, the root dbspace), the database server re-creates the temporary dbspaces but they are empty.*

What Is a Warm Restore?

The database server is in online, quiescent, or fast-recovery mode in a warm restore. Unless your database server has failed, you can restore noncritical storage spaces in a warm restore in the following circumstances:

- The storage space is online, but one of its chunks is offline, recovering, or inconsistent.
- The storage space is offline or down.

If the database server goes offline but the critical dbspaces are all right, bring the database server online and perform a warm restore. If the database server goes offline and a critical dbspace is down, perform a cold restore. For details, see [“What Is a Cold Restore?” on page 6-4](#).

What Is a Cold Restore?

If a critical dbspace is damaged because of a disk failure or corrupted data, the database server goes offline automatically. If a critical dbspace goes down, you must perform a *cold restore* of all critical dbspaces.

The database server must be offline on Dynamic Server or in microkernel mode on Extended Parallel Server for a cold restore. You can perform a cold restore of all storage spaces regardless of whether they were online or offline when the database server went down.

Perform a cold restore when the database server fails or you need to perform one of the following tasks:

- Point in time restore
- Point in log restore
- Whole system restore (IDS only)
- Imported restore
- Renaming chunks (IDS only)

A cold restore starts by physically restoring all critical storage spaces, then the noncritical storage spaces, and finally the logical logs. The database server goes into recovery mode after the reserved pages of the root dbspace are restored. When the logical restore is complete, the database server goes into quiescent mode. Use the **onmode** command to bring the database server online. For more information, see [“Performing a Cold Restore” on page 6-19](#).



Tip: *If you mirror the critical dbspaces, you are less likely to have to perform a cold restore after a disk failure because the database server can use the mirrored storage space. If you mirror the logical-log spaces, you are more likely to be able to salvage logical-log data if one or more disks fail.*

What Is a Mixed Restore?

A *mixed restore* is a cold restore of some storage spaces followed by a warm restore of the rest of the storage spaces. If you need to provide access to a particular table or set of tables as soon as possible, you might want to perform a mixed restore. A mixed restore restores the critical dbspaces first, then the other storage spaces that you specify during a cold restore. After the database server is back online, perform one or more warm restores of the remaining storage spaces and logical logs.

The storage spaces that you do not restore during the cold restore are not available until after you restore them during a warm restore, even though they might not have been damaged by the failure. While a mixed restore makes the critical data available sooner, the complete restore takes longer because the logical logs are restored twice, once during the cold restore and again during the warm restore.

What Is a Parallel Restore?

If you perform a restore using the **onbar -r** command, ON-Bar restores the storage spaces in parallel and replays the logical logs once.

If BAR_MAX_BACKUP is set to 1, ON-Bar restores the storage spaces serially. If you did not perform a whole-system backup, you must use the **onbar -r** command to restore the data. ♦

If BAR_WORKER_MAX is set to 1, ON-Bar restores the storage spaces serially. ♦

What Is a Whole-System Restore?

Whole-system restores are always serial. If you backed up the whole system (**onbar -b -w**), you can restore the whole system (**onbar -r -w** or **onbar -r -p -w**) with or without restoring the logical logs. Choose a whole-system restore for small systems or when you do not need to restore logs. You can perform a whole-system point-in-time restore.

IDS

XPS

IDS

What Is a Point-in-Time Restore?

A point-in-time restore is a cold restore that you can use to undo mistakes that might otherwise not be fixable. An example of such a mistake is dropping a table by mistake. A full restore restores the table during the physical restore but drops it again during the logical restore. A point-in-time restore lets you restore the data to the moment just before the table was dropped.

When you restore the database server to a specific time, any transactions that were uncommitted at the specified point in time are lost. Also, all transactions after the point-in-time restore are lost. For information on how to restore to a specific time, see [“Restoring Data to a Point in Time” on page 6-22](#).

What Is an Imported Restore?

ON-Bar allows you to restore objects to a different database server instance than the one from which the data was backed up. This type of restore is called an *imported restore*.

You can perform imported restores using whole-system, serial, or parallel backups. You must use compatible versions of XBSA and storage managers for both operations. If you perform a parallel imported restore, it must include all storage spaces, logical logs, and administrative files from the source database server to synchronize the instance. For more information, see [“Transferring Data with the Imported Restore” on page 6-32](#).

You cannot use a backup from one database server version to restore on a different version.

What is a Rename Chunks Restore?

ON-Bar allows you to rename chunks by specifying new chunks paths and offsets during a cold restore. This option is useful if you need to restore storage spaces to a different disk from the one on which the backup was made. You can rename any type of chunk, including critical chunks and mirror chunks.

For more information, see [“Renaming Chunks During a Restore”](#) on [page 6-27](#).

What Is a Restartable Restore?

If a failure occurs with the database server, media, or ON-Bar during a restore, you can restart the restore from the place that it failed. You do not have to restart the restore from the beginning. The `RESTARTABLE_RESTORE` parameter controls whether ON-Bar is able to keep track of the storage spaces and logical logs that were restored successfully.

You can restart the following types of restores:

- Whole system
- Point in time
- Storage spaces
- Logical part of a cold restore

For more details, see [“Using Restartable Restore to Recover Data”](#) on [page 6-39](#) and [“RESTARTABLE_RESTORE”](#) on [page 9-27](#).

Using the Pre-Recovery Checklist

Use this checklist before you decide if a restore is required:

- Has data been lost or corrupted?
- Does a committed transaction error need to be undone?
- Is the database server down or has a disk failed?
- Is a storage space or chunk down or inconsistent?

- Review the following files and outputs to obtain information about your system:
 - ❑ The **onstat -d** and **onstat -l** outputs
 - ❑ The database server message log
 - ❑ The ON-Bar activity log
 - ❑ The storage-manager logs
 - ❑ The **oncheck** output (Dynamic Server) or **onutil** CHECK output (Extended Parallel Server)
 - ❑ The **oncfg** files
 - ❑ The physical data layout (disks)
 - ❑ The database schema (**dbschema** command)
 - ❑ The **af*** files (assertion failures), if any
 - ❑ The core dump files, if any
 - ❑ The **xcfg** file ♦
 - ❑ The **ism_chk.pl** report

The **ism_chk.pl** report is useful when you investigate backup or restore problems. For details, see the *IBM Informix Storage Manager Administrator's Guide*.

- Estimate how long the restore will take.
- Determine whether a warm or cold restore is needed. If you need to take the database server offline for the restore, ask your client users to log off the system.
- If you suspect a problem with the storage manager or the XBSA connection, the operating system, or the storage media, contact your vendor.

Monitoring Restores

To determine the state of each storage space and its chunks, or the status of the restore, examine the output of the **onstat -d** utility. The **onstat -d** utility works only with the database server online. For more information on **onstat -d**, see “[onstat -d](#)” on page B-2. The following table describes the information in the second position of the **flags** column in the first (storage spaces) and second (chunks) sections of the **onstat -d** output and the actions required to solve the problems.

XPS

On Extended Parallel Server, you can use the `xctl onstat -d` utility to check the storage spaces on all coservers. ♦

onstat -d Flag	Storage Space or Chunk State	Action Required
	Storage space no longer exists.	Perform a point-in-time cold restore to a time before the storage space was dropped.
D	Chunk is down or storage space is disabled.	Perform a warm restore of the affected storage space.
I	Chunk has been physically restored, but needs a logical restore.	Perform a logical restore.
L	Storage space is being logically restored.	Retry the logical restore.
N	Chunk is renamed and either down or inconsistent. (IDS)	Perform a warm restore of the chunk when the physical device is available.
O	Chunk is online.	No action required.
P	Storage space is physically restored.	Perform a logical restore, if one is not already in progress.
R	Storage space is being restored.	Perform a physical or logical restore.
X	Storage space or chunk is newly mirrored.	No action required.

Ensuring That Storage Devices Are Available

Verify that storage devices and files are available before you begin a restore. For example, when you drop a dbspace or mirroring for a dbspace after your level-0 backup, you must ensure that the dbspace or mirror chunk device is available to the database server when you begin the restore. If the storage device is not available, the database server cannot write to the chunk and the restore fails.

When you add a chunk after your last backup, you must ensure that the chunk device is available to the database server when it rolls forward the logical log.

Restoring Save Sets with ISM

If you are using ISM, you can restore data from save sets on the storage volume. When the ISM server receives a restore request, the **ism_watch** command prompts you to mount the required storage volume on the storage device. When you mount the volume, the restore will resume.

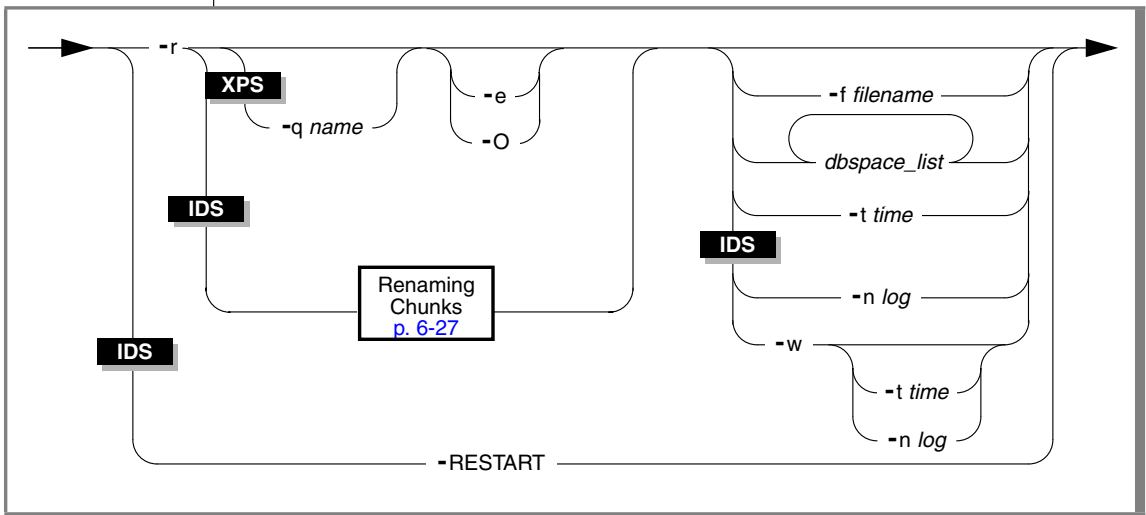
You can set the retention period for either a save set or volume. Unless all the save sets on the volume have expired, you can use ON-Bar to restore it.

After the retention period for a save set expires, ON-Bar can no longer restore it. To re-create an expired save set, use the **ism_catalog -recreate from** command.

If you set the retention period for a volume, ISM retains the save sets until all the save sets on that volume have expired. To recover an expired volume, use the **ism_catalog -recover from** command. For more information, see the *IBM Informix Storage Manager Administrator's Guide*.

Performing a Complete Restore

This diagram shows the syntax for the **onbar -r** commands.



Element	Purpose	Key Considerations
-r	Specifies a restore	In a cold restore, the -r option restores all storage spaces, salvages and restores the logical logs. In a warm restore, the -r option restores all offline storage spaces and restores the logical logs. You must specify the -r parameter first.
dbspace_list	Names one or more dbspaces, blobspaces (IDS), sbspaces (IDS), or dbslices (XPS) to be restored	ON-Bar restores only the storage spaces listed. If it is a cold restore, you must list all the critical dbspaces. If you enter more than one storage-space name, use a space to separate the names.
-e	Specifies an external restore	After you externally restore the storage spaces with a third-party utility, run onbar -r -e to mark the storage spaces as physically restored, restore the logical logs, and bring the storage spaces back online. For details, see “Using External Restore Commands” on page 7-17 .
-f filename	Restores the storage spaces that are listed in the text file whose pathname <i>filename</i> provides Use this option to avoid entering a long list of storage spaces every time that you use this option.	The filename can be any valid UNIX or Windows filename, including simple (listfile_1), relative (../backup_lists/listfile_2 or ..\backup_lists\listfile2), and absolute (/usr/informix/backup_lists/listfile3 or c:\informix\backup_lists\listfile3) filenames. This file can list multiple storage spaces per line.
-n log	Indicates the <i>uniqid</i> of the last log to restore in a cold restore To find the <i>uniqid</i> number, use the onstat -l command (IDS).	A point-in-log restore is a special kind of point-in-time restore. You must restore all storage spaces in a point-in-log restore so that the data is consistent. If any logical logs exist after this one, ON-Bar does not restore them and their data is lost.
-O	Allows a restore of online storage spaces This option also re-creates missing chunk files.	If a chunk file no longer exists, the -O option lets ON-Bar re-create it. This new chunk file is cooked disk space, not raw disk space. If a storage space in the list of spaces to restore is online, the -O option lets ON-Bar bring it offline and then restore it. <ul style="list-style-type: none"> ■ If an online storage space is restored successfully, return code 177 displays in the ON-Bar activity log. ■ If a chunk is re-created successfully, return code 179 displays in the ON-Bar activity log. Use the -O option with a whole-system restore to re-create missing chunk files (IDS).

(1 of 2)

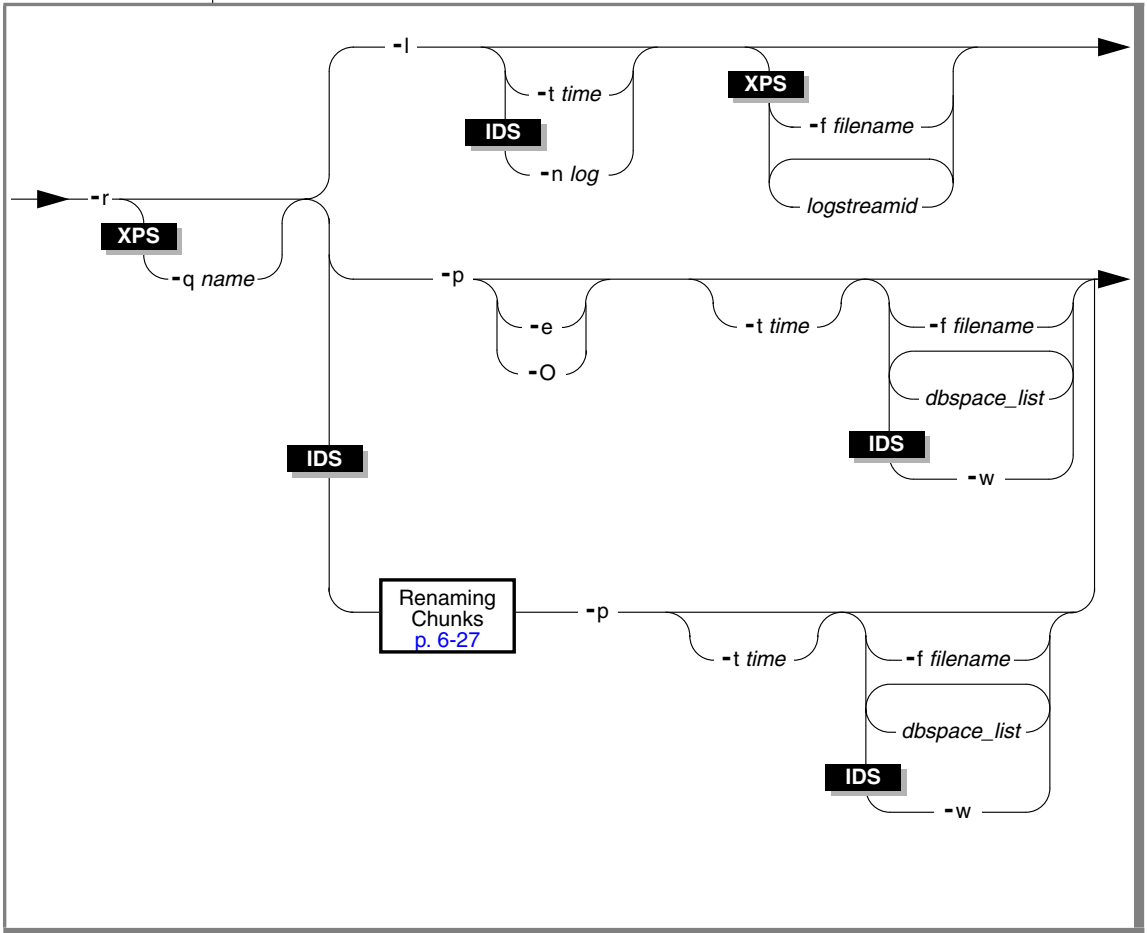
Performing a Complete Restore

Element	Purpose	Key Considerations
-q name	Allows you to assign a name to the restore This name appears in the onstat utility so that you can follow the progress of the backup session (XPS).	<i>DBSERVERNAME</i> random_number is the default session name. The session name must be unique and can be up to 128 characters.
-RESTART	Restarts a restore after a database server or ON-Bar failure (IDS)	For the restore to be restartable, the RESTARTABLE_RESTORE configuration parameter must be set to ON when the restore failure occurs. If RESTARTABLE_RESTORE is set to OFF , the -RESTART option does not work. For more information, see “Using Restartable Restore to Recover Data” on page 6-39 .
-t time	Specifies the time of the last transaction to be restored from the logical logs in a cold restore	You must specify the onbar -r -t option (point-in-time) in a cold restore only and must restore all storage spaces to the same time. For more information, see “Restoring Data to a Point in Time” on page 6-22 .
-w	Performs a whole-system restore of all storage spaces and logical logs from the last whole-system backup (IDS)	You must specify the -w option in a cold restore. If you specify onbar -r -w without a whole-system backup, return code 147 appears because ON-Bar cannot find any storage spaces backed up as part of a whole-system backup.

(2 of 2)

Performing a Physical-Only or Logical-Only Restore

This diagram shows the syntax for a physical-only or logical-only restore.



Performing a Physical-Only or Logical-Only Restore

Element	Purpose	Key Considerations
-r	If specified with the -p option, restores the storage spaces only If specified with the -l option, restores the logical logs only.	You must specify the -r parameter first.
dbspace_list	Names one or more dbspaces, blobspaces (IDS), sbspaces (IDS), or dbslices (XPS) to be restored	ON-Bar restores only the storage spaces listed. If it is a cold restore, you must list all the critical dbspaces. If you enter more than one storage-space name, use a space to separate the names.
-e	Specifies an external restore	After you externally restore the storage spaces with a third-party utility, run onbar -r -e to mark the storage spaces as physically restored, restore the logical logs, and bring the storage spaces back online. For details, see “Using External Restore Commands” on page 7-17 .
-f filename	Restores the storage spaces that are listed in the text file whose pathname <i>filename</i> provides Use this option to avoid entering a long list of storage spaces every time that you use this option.	The filename can be any valid UNIX or Windows filename, including simple (listfile_1), relative (../backup_lists/listfile_2 or ..\backup_lists\listfile2), and absolute (/usr/informix/backup_lists/listfile3 or c:\informix\backup_lists\listfile3) filenames. This file can list multiple storage spaces per line.
-l	Specifies a logical restore only Restores and rolls forward the logical logs.	The logical restore applies only to those storage spaces that have already been physically restored.
logstreamid	Uniquely identifies logical-log records that a given coserver generates (XPS)	If you supply more than one <i>logstreamid</i> , separate each item in the list with a space. A logstream is a coserver ID.
-n log	Indicates the <i>uniqid</i> of the last log to restore in a cold restore To find the uniqid number, use the onstat -l command (IDS).	A point-in-log restore is a special kind of point-in-time restore. You must restore all storage spaces in a point-in-log restore so that the data is consistent. If any logical logs exist after this one, ON-Bar does not restore them and their data is lost.

(1 of 2)

Element	Purpose	Key Considerations
-O	Allows a restore of online storage spaces This option also re-creates missing chunk files.	<p>If a chunk file no longer exists, the -O option lets ON-Bar re-create it. This new chunk file is cooked disk space, not raw disk space.</p> <p>If a storage space in the list of spaces to restore is online, the -O option lets ON-Bar bring it offline and then restore it.</p> <ul style="list-style-type: none"> ■ If an online storage space is restored successfully, return code 177 displays in the ON-Bar activity log. ■ If a chunk is re-created successfully, return code 179 displays in the ON-Bar activity log. <p>Use the -O option with a whole-system restore to re-create missing chunk files (IDS).</p>
-p	Specifies a physical restore only	You must follow a physical restore with a logical restore before data is accessible unless you use a whole-system restore. This option turns off automatic log salvage before a cold restore.
-q name	Allows you to assign a name to the restore This name appears in the onstat utility so that you can follow the progress of the backup session (XPS).	<i>DBSERVERNAME</i> random_number is the default session name. The session name must be unique and can be up to 128 characters.
-t time	Specifies the time of the last transaction to be restored from the logical logs in a cold restore	<p>You can specify the onbar -r -p -t command in a warm or cold restore to restore specific storage spaces from an old physical backup. You must then use onbar -r -l to finish the logical restore.</p> <p>For more information, see “Restoring from an Older Backup” on page 6-22.</p>
-w	Performs a whole-system restore (IDS)	To restore the storage spaces only, specify the -w -p option in a cold restore.

(2 of 2)

Using IBM Informix Server Administrator to Restore Data

You can use IBM Informix Server Administrator (ISA) to perform backups and restores with ON-Bar. For more information, see the ISA online help.

Examples of ON-Bar Restore Commands

The following sections contain examples of ON-Bar syntax for restoring data.

For an example of renaming chunks during a cold restore, see [“Renaming Chunks During a Restore” on page 6-27](#).

Performing a Restore

To perform a complete cold or warm restore, use the **onbar -r** command. ON-Bar restores the storage spaces in parallel. To speed up restores, you can add additional CPU virtual processors. To perform a restore, use the following command:

```
onbar -r
```

In a warm restore, the **-r** option restores all down storage spaces and logical logs, and skips online storage spaces. A *down* storage space means it is offline or a chunk in it is inconsistent.

In a cold restore, the **-r** option automatically salvages the logical logs, and restores all storage spaces and appropriate logical logs.



Tip: For faster performance in a restore, assign separate storage devices for backing up storage spaces and logical logs. If physical and logical backups are mixed together on the storage media, it takes longer to scan the media during a restore.

Restoring Specified Storage Spaces

To restore particular storage spaces (for example, a dbspaces named `fin_dbspace1` and `fin_dbspace2`), use the **-r** option, as the following example shows:

```
onbar -r fin_dbspace1 fin_dbspace2
```

If any named dbspaces are online, they are skipped in the restore. ON-Bar writes a message to the activity log about the skipped dbspaces.

Performing a Logical Restore

To perform a logical restore, use the following command:

```
onbar -r -l
```



Warning: Because the logical-log files are replayed using temporary space during a warm restore, make sure that you have enough temporary space for the logical restore.

The minimum amount of temporary space that the database server needs is equal to:

- The total logical-log space for the database server instance, or the number of log files to be replayed, whichever is smaller. ♦
- The total logical-log space for the coservers on which storage spaces are being restored, or the number of log files to be replayed, whichever is smaller. Each coserver must have enough temporary space for all its temporary log files. ♦



Tip: To improve performance, replay logical-log transactions in parallel during a warm restore. Use the `ON_RECOVERY_THREADS` configuration parameter to set the number of parallel threads. To replay logical-log transactions in parallel during a cold restore, use the `OFF_RECOVERY_THREADS` configuration parameter. For more information, see your “Performance Guide.”

IDS

XPS

XPS

Restoring Data on Extended Parallel Server

To restore all dbspaces in a dbslice named `fin_slice`, use the following command:

```
onbar -r fin_slice
```

If you have a mixture of dbspaces to restore, some that require logical replay and some that do not, follow these steps:

1. Restore the dbspaces that do not require logical replay first.
2. Restore the dbspaces that require logical replay.

You can use the dbspaces restored in the first pass sooner but the total restore time might be longer. This method enables you to quickly restore tables in a data warehouse. For more information, see [“Skipping Logical Replay” on page 4-29](#).

Performing a Physical Restore Followed By a Logical Restore

In certain situations, you might want to perform a restore in stages. The combination of physical and logical restores ensures that tables and indexes are as current as possible. If multiple devices are available for the restore, you can restore multiple storage spaces separately or concurrently, and then perform a single logical restore.

To perform a warm restore in stages

1. Perform a physical restore:

```
onbar -r -p
```

2. Back up the logical logs:

```
onbar -b -l
```

3. Perform a logical restore:

```
onbar -r -l
```

To perform a cold restore in stages

1. Optionally, salvage the logical logs manually:

```
onbar -b -l -s
```

To perform a cold restore without salvaging the logical logs, skip this step.

2. Perform a physical restore:

```
onbar -r -p
```

3. Perform a logical restore:

```
onbar -r -l
```

4. Synchronize the **sysutils** database and emergency boot files:

```
onmsmsync
```

You must run **onmsmsync** after performing a cold restore in stages, but do not need to run **onmsmsync** after a complete cold restore (**onbar -r**).

For information on what actions to take when an error occurs during a physical or logical restore, see [“Resolving a Failed Restore” on page 6-44](#).

IDS

Salvaging Logical Logs

Decide whether you want to salvage the logical logs before you perform a cold restore. If not, the data in the logical logs that has not been backed up is lost. If a disk is damaged, salvage the logs if they are still accessible before you replace the disk. For more information, see [“Performing a Cold Restore” on page 6-19](#).

The **onbar -r** command automatically salvages the logical logs. Use the **onbar -r -p** and **onbar -r -l** commands if you want to skip log salvage.

If you set the LTAPEDEV configuration parameter to **/dev/null** on UNIX or to **NUL** or **\dev\nul** on Windows, the logical logs are *not* salvaged in any ON-Bar restore (**onbar -r** or **onbar -r -w**, for example). ♦

Avoid salvaging the logical logs in the following situations:

- When you perform an imported restore
Salvage the logical logs on the source database server but not on the target database server. For more information, see [“Transferring Data with the Imported Restore” on page 6-32](#).
- If you reinitialize the database server (**oninit -i**) before you perform a cold restore
Reinitialization creates new logical logs that do not contain the data that you want to restore.
- If you install a new disk for the dbspace that contains the logical logs
Salvage the logs from the old disk, but not from the new disk.

Performing a Cold Restore

If a critical storage space is damaged because of a disk failure or corrupted data, you must perform a cold restore. If a disk fails, you need to replace it before you can perform a cold restore to recover data.

On Extended Parallel Server, if a critical dbspace on any of the coservers goes down, you must perform a cold restore on all coservers. ♦

Warning: *Back up all storage spaces before you perform a cold restore. If not and you try a cold restore, data in the storage spaces that were not backed up will be lost. The storage space is marked as offline after the cold restore. Drop the storage space so that you can reuse its disk space.*

XPS



IDS

XPS

To perform a cold restore with automatic log salvage

1. Copy the administrative files (ONCONFIG, **sqlhosts** [UNIX only], emergency boot files, **oncfg** files, and **xcfg** files [XPS only]) to a safe place.

2. Take the database server offline with the following command:

```
onmode -ky
```

◆

Take the database server offline and then bring it to microkernel mode:

```
xctl onmode -ky  
xctl -C oninit -m
```

◆

3. If the disk that contains the logical-log files needs to be replaced or repaired, use the following command to salvage logical-log files on the damaged disk:

```
onbar -b -l -s
```

4. Then repair or replace the disk.

5. If the files in **INFORMIXDIR** were destroyed, copy the previously saved administrative files to their original locations.

However, if you did the cold restore because a critical dbspace was lost, you do not need to copy the administrative files. For more information, see [“Which Administrative Files to Back Up?” on page 4-6](#).

6. To restore the critical and noncritical storage spaces, use the following command:

```
onbar -r
```

When the restore is complete, the database server is in quiescent mode.

7. To bring the database server online, use the following command:

```
onmode -m
```

Performing a Whole-System Restore

A whole-system restore must be cold and must restore all storage spaces. A whole-system restore does not require you to restore the logical logs.

A whole-system restore requires a whole-system backup. However, you can perform a plain restore of a whole-system backup. If you use **onbar -b -w** to back up the whole system, you can restore with any of the following commands:

```
onbar -r -w          # whole-system restore (salvages logs automatically)
onbar -r -p -w       # physical-only whole-system restore (no log salvage)
onbar -r             # parallel restore of the whole-system backup
onbar -r dbspaces     # restore dbspaces from a whole-system backup
onbar -r -t time      # point-in-time restore
onbar -r -t time -w   # whole-system point-in-time restore
```

If you use **onbar -r -p -w**, the database server is in fast recovery mode when the restore completes. Perform either a logical restore (**onbar -r -l**) or use **onmode -m** to bring the database server online. For more information, see [“Performing a Whole-System Backup” on page 4-18](#).

Considerations When LTAPEDEV is Set to Null

A whole-system backup with LTAPEDEV set to **/dev/null** on UNIX or to **\\dev\\nul** on Windows does not back up the logical logs.

To restore the data from a whole-system backup when LTAPEDEV is null

1. Upon restore, you must use the **onbar -r -w -p** command.
When the physical-only whole-system restore completes, the database server is in fast recovery mode.
2. If the database server is offline, use the **onmode -sy** command to perform fast recovery.
If the database server is online, use the **onmode -m** command to perform fast recovery.

Using the -O Option in a Whole-System Restore

Use the **-O** option with a whole-system restore only to re-create missing chunk files. You cannot use the **onbar -r -w -O** command when the database server is online because the root dbspace cannot be taken offline during the whole-system restore.

Restoring Data to a Point in Time

Perform a point-in-time restore if you do not want to replay a mistake that was recorded in the logical logs. If you use the **onbar -r -t time** command, you must restore *all* storage spaces to a specific time in a cold restore.

To restore database server data to its state at a specific date and time, enter a command using the date and time format for your GLS locale, as this example shows:

```
onbar -r -t "1997-05-10 12:00:00"
```

Quotes are recommended around the date and time. The format for the English locale is **yyyy-mm-dd hh:mm:ss**. If the **GL_DATETIME** environment variable is set, you must specify the date and time according to that variable. For an overview, see [“What Is an Imported Restore?” on page 6-6](#). For an example of using a point-in-time restore in a non-English locale, see [“Using GLS with ON-Bar” on page D-1](#).



Important: To determine the appropriate date and time for the point-in-time restore, use the **onlog** utility that the “Administrator’s Reference” describes. The **onlog** output displays the date and time of the committed transactions in the logical log. All data transactions that occur after **time** or **last_log** are lost.

You can also perform a whole-system, point-in-time restore. ♦

Restoring from an Older Backup

By default, ON-Bar restores the latest backup. If you do not want to restore this backup (for example, when backup verification failed or the backup media was lost), you can restore from an older backup.

To restore from an older backup using a physical point-in-time restore

1. Find the time of the older backup in the message log or ON-Bar activity log or from the storage manager.
2. To restore all or specific storage spaces, issue the following commands:

```
onbar -r -p -t time [dbspaces_from_older_backup]
onbar -r -p [remaining_dbspaces]
onbar -r -l
```

IDS

IDS

To restore from an older backup by expiring the bad backup

1. Expire the bad backup in the storage manager.
2. Run **onsmsync**.
3. To restore the data, issue the following command:

```
onbar -r
```

Performing a Point-in-Log Restore

A point-in-log restore is similar to a point-in-time restore. The point-in-log restore stops at the time of the last committed transaction listed in the logical log. You must use point-in-log restore in a cold restore only and must restore all storage spaces. To perform a point-in-log restore, use the following command:

```
onbar -r -n last_log
```

Restoring Online Storage Spaces

Use the following command to force a restore of online storage spaces (except critical dbspaces) in a warm restore:

```
onbar -r -O dbsp1 dbsp2
```

The database server automatically shuts down each storage space before it starts to restore it. Taking the storage space offline ensures that users do not try to update its tables during the restore process.

IDS

For special considerations on using the **-O** option, see [“Using the -O Option in a Whole-System Restore” on page 6-21](#). ♦

Re-Creating Chunk Files During a Restore

If the disk or file system fails, one or more chunk files could be missing from the dbospace. If you use the **-O** option in a warm or cold restore, ON-Bar re-creates the missing chunk files, including any necessary directories, before restoring the dbospace as long as enough space exists on the file system. The newly created chunk files are cooked files and are owned by group **informix** on UNIX or group **Informix-Admin** on Windows.

To restore when using cooked chunks

1. Install the new disk.
2. On UNIX, mount the device as a file system.
On Windows, format the disk.
3. Allocate disk space for the chunk file.
For instructions, see the chapter on managing data in the *Administrator's Guide*.
4. Issue the following command to re-create the chunk files and restore the dbospace:

```
onbar -r -O crashedspace
```

Important: ON-Bar does not re-create chunk files during a logical restore if the logical logs contain chunk-creation records.

To restore when using raw chunks

1. Install the new disk.
2. If you use symbolic links to raw devices, create new links for the down chunks that point to the newly installed disk.
ON-Bar restores the chunk file where the symbolic link points. ♦
3. Issue the following command to restore the dbospace:

```
onbar -r crashedspace
```



UNIX

Restoring a Dropped Storage Space

If you accidentally drop a storage space, you can use a point-in-time restore or a point-in-log restore to recover it.

To restore a dropped storage space using a point-in-time restore

1. In this example, the database server has **dbspace1**, which was dropped accidentally, and **dbspace2**. Use **onlog** or the database server message log to obtain a time before **dbspace1** was dropped.
2. Shut down the database server.
3. Perform a point-in-time restore:

```
onbar -r -t time
```

This command restores the **rootdbs** and **dbspace2** but fails with error 131 and the following message: “Unable to start the logical-log restore: A point-in-time logical restore is only permitted during a full restore. Dbspace *dbspace1* was not recovered during this restore.” Ignore this error message.

4. To restore the dropped storage space, enter the following command:

```
onbar -r -t time dbspace1
```

To restore a dropped storage space using separate physical and logical restores

1. In this example, the database server has **dbspace1**, which was dropped accidentally, and **dbspace2**. Use **onlog** or the database server message log to obtain a time before **dbspace1** was dropped.
2. Shut down the database server.
3. Perform a physical-only restore of all storage spaces:

```
onbar -r -p -t time rootdbs dbspace1 dbspace2
```

4. To restore the dropped storage space and prevent the logical log from replaying the drop, enter one of the following commands:

- a. If you use the point-in-log command, specify the *uniqid* of the log before the log that contains the drop command:

```
onbar -r -l -n uniqid
```

- b. If you use the logical point-in-time command, use the same time as in step 3:

```
onbar -r -l -t time
```



To restore a dropped storage space when the chunk files were also deleted

1. Use the **onlog** utility to find the logical-log file that contains the dropped transaction for the storage space.
2. To restore a dropped storage space when the chunk files were deleted, enter the following command:

```
onbar -r -t time -O
```

The point-in-time restore restores the dropped storage space and automatically re-creates the chunk files.

Warning: You must restore the data to a point in time before the storage space was dropped in both the physical and logical restores.

Restoring Data when Reinitializing the Database Server

Any backups that you performed before reinitializing the database server are unusable. During initialization, ON-Bar saves the emergency boot file elsewhere and starts a new, empty emergency boot file. Do not use the copy of the emergency boot file unless you want to restore the previous database server instance.

To reinitialize the database server after a failure when you do not need the old data

1. Do not copy the old emergency boot file into the database server directory (\$INFORMIXDIR/etc on UNIX or %INFORMIXDIR%\etc on Windows).
2. To perform a complete backup, use **onbar -b**.

To reinitialize the database server and restore the old data

1. Before you reinitialize the database server, copy the administrative files (emergency boot, **oncfg**, and ONCONFIG files) to a different directory, if possible.
2. Reinitialize the database server.
3. Re-copy the administrative files into the database server directory because you need the information in the old emergency boot file.
If the administrative files are unavailable, copy them from the last backup into the database server directory.
4. Perform a restore. Do not salvage the logical logs.



5. Any changes made after reinitialization are lost.
6. Verify that you restored the correct instance of the critical and noncritical storage spaces.

Renaming Chunks During a Restore

You can rename chunks by specifying new chunks paths and offsets during a cold restore with ON-Bar. This option is useful if you need to restore storage spaces to a different disk from the one on which the backup was made. You can rename any type of chunk, including critical chunks and mirror chunks.

Tip: *If you use symbolic links to chunk names, you might not need to rename chunks; you need only edit the symbolic name definitions. For more information, see the “Administrator’s Guide.”*

Key Considerations

During a cold restore, ON-Bar performs the following validations to rename chunks:

1. It validates that the old chunk pathnames and offsets exist in the archive reserved pages.
2. It validates that the new chunk pathnames and offsets do not overlap each other or existing chunks.
3. If renaming the primary root or mirror root chunk, it updates the ONCONFIG file parameters ROOTPATH and ROOTOFFSET, or MIRRORPATH, and MIRROROFFSET. The old version of the ONCONFIG file is saved as **\$ONCONFIG.localtime**.
4. It restores the data from the old chunks to the new chunks (if the new chunks exist).
5. It writes the rename information for each chunk to the online log.

If either of the validation steps fails, the renaming process stops and ON-Bar writes an error message to the ON-Bar activity log.

Warning: *Perform a level-0 archive after you rename chunks; otherwise you will have to specify chunk mappings again during the next restore.*





Important: If you add a chunk after performing a level-0 archive, that chunk cannot be renamed during a restore. Also, you cannot safely specify that chunk as a new path in the mapping list.

Important: Renaming chunks for database servers participating in HDR involves a significant amount of time offline for both database servers. For more information, see the “Administrator’s Guide.”

New-Chunk Requirements

To rename a chunk, follow these guidelines for new chunks:

- The new chunk does not need to exist

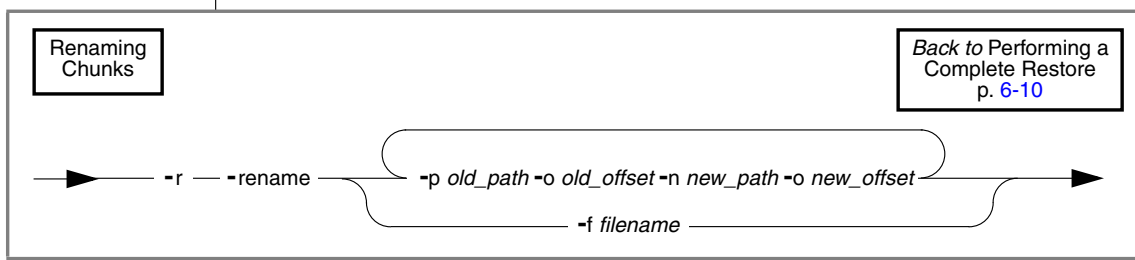
You can install the new chunk later and perform a warm restore of a storage space containing it. If you specify a nonexistent chunk, ON-Bar records the rename information in the chunk reserved pages, but does not restore the data. The renamed (but not restored) chunks have a status of offline, designated by **N**, in the **onstat -d** chunk status command output.

- New chunks must have the proper permissions.

Rename operations fail unless the chunks have the proper permissions. For more information, see the *Administrator’s Guide*.

Syntax

This diagram shows the ON-Bar syntax for renaming chunks during a cold restore.



Element	Purpose	Key Considerations
-r	Specifies a restore	You must specify the -r parameter first.
-rename	Renames one or more chunks during a cold restore	
-f filename	Specifies a file containing the names and offsets of chunks to be renamed and their new locations Use to rename a large number of chunks at one time	The filename can be any valid UNIX or Windows filename, including simple (listfile_1), relative (../backup_lists/listfile_2 or ..\backup_lists\listfile2), and absolute (/usr/informix/backup_lists/listfile3 or c:\informix\backup_lists\listfile3) filenames. In the file, list the old chunk pathname, the old offset, the new chunk pathname, and the new offset. Put a blank space or a tab between each item. Put information for each chunk on a separate line. Blank lines are ignored. Begin comment lines with a # symbol.
-p old_path -o old_offset -n new_path -o new_offset	Specifies the chunk to be renamed and its new location Use to rename one or more chunks at one time	The variables for this element are: <ul style="list-style-type: none">■ old_path is the current path and filename of the chunk■ old_offset is the current offset of the chunk, in kilobytes■ new_path is the new path and filename of the chunk■ new_offset is the new offset of the chunk

You can use the following options after the **-rename** command:

- **-f filename**
- **dbspace_list**
- **-t time**
- **-n log**
- **-w**
- **-p**

For more information on these options, see [“Performing a Complete Restore” on page 6-10](#).

Examples of Renaming Chunks During a Restore

To rename a chunk, provide the old chunk location and the new chunk location, either at the command line or in a file.

The following table lists example values for two chunks that are used in the examples in this section.

Element	Value for First Chunk	Value for Second Chunk
old path	/chunk1	/chunk2
old offset	0	10000
new path	/chunk1N	/chunk2N
new offset	20000	0

Renaming Chunks with Command Line Options

To rename the chunks by supplying information on the command line, use this command:

```
onbar -r -rename -p /chunk1 -o 0 -n /chunk1N -o 20000
        -rename -p /chunk2 -o 10000 -n /chunk2N -o 0
```

Perform a level-0 archive after the rename and restore operation is complete.

Renaming Chunks with a File

To rename the chunks by supplying a file named **listfile**, use this command:

```
onbar -r -rename -f listfile
```

The contents of the **listfile** file are:

```
/chunk1 0 /chunk1N 20000
/chunk2 10000 /chunk2N 0
```

Perform a level-0 archive after the rename and restore operation is complete.

Renaming Chunks While Specifying Other Options

To rename the chunks using command-line options while performing a physical restore on **dbspace1** and **dbspace2**, use the following command:

```
onbar -r -rename -p /chunk1 -o 0 -n /chunk1N -o 20000
        -rename -p /chunk2 -o 10000 -n /chunk2N -o 0
        -p dbspace1 dbspace2
```

Alternatively, to rename the chunks using file while performing a physical restore on **dbspace1** and **dbspace2**, use the following command:

```
onbar -r -rename -f listfile -p dbspace1 dbspace2
```

Perform a level-0 archive after the rename and restore operation is complete.

Renaming a Chunk to a Nonexistent Device

To rename a chunk to a device that does not yet exist, you specify the new path name, but you do not restore its storage spaces until after you install the physical device. This option is useful if you need to rename a chunk and it is convenient to perform a cold restore before you install the new device. When the new chunk device is ready, you can perform a warm restore of a storage space onto it.

You can combine renaming chunks with existing devices and renaming chunks with nonexistent devices in the same rename operation. This example shows how to rename a single chunk to a nonexistent device name.

The following table lists example values for the chunks used in this example.

Storage Space	Old Chunk Path	Old Offset	New Chunk Path	New Offset
sbspace1	/chunk3	0	/chunk3N	0

To rename a chunk to a nonexistent device

1. Rename the chunk:

```
onbar -r -rename -p /chunk3 -o 0 -n /chunk3N -o 0
```

2. When the following prompt appears, enter **y** to continue: The chunk /chunk3N does not exist. If you continue, the restore may fail later for the dbspace which contains this chunk. Continue without creating this chunk? (y/n)

The chunk **/chunk3** is renamed to **/chunk3N**, but the data has not yet been restored to **/chunk3N**.

3. Perform a level-0 archive.
4. Add the physical device for **/chunk3N**.
5. Perform a warm restore of **sbospace1**:

```
onbar -r sbospace1
```

6. Perform a level-0 archive.

Transferring Data with the Imported Restore

With the *imported restore* feature, you can transfer all the data from one instance of the database server to the same instance on a foreign host. For example, you can back up data on one computer and restore the data on a different computer. You can perform imported restores using whole-system, parallel, or serial backups.

The imported restore is useful in the following situations:

- Disaster recovery
- Database server upgrade
- Initialization of High-Availability Data Replication (HDR)

When you prepare for an imported restore, consider these points:

- Make sure that your storage manager supports imported restores.
- The whole-system backup must include all storage spaces; logical logs are optional.
The parallel backup must include all storage spaces and logical logs.
- You can change the database server name in an imported restore.



Important: You cannot use a backup from one database server version to restore on a different version.

For information on importing a restore with ISM, see the *IBM Informix Storage Manager Administrator's Guide*. For information on using HDR, see the *Administrator's Guide*. If you are using a third-party storage manager, use the following procedure for an imported restore.

To set up an imported restore

1. Install the database server and the storage manager on the target computer. Both computers must have the following:
 - Identical hardware and operating systems
 - Identical database server versions
 - Be on the same LAN or WAN
 - Identical storage-manager versions
 - Compatible XBSA libraries

The source computer (also called the *primary server*) contains the current instance that you want to replicate. The target computer (also called the *secondary server*) contains the computer where you want to replicate the source instance.

2. Set up the storage manager on the target database server instance.
 - a. Define the same type of storage devices as on the source instance.
 - b. Label the storage media with the correct pool names.
 - c. Mount the storage devices.
 - d. Update the **sm_versions** file on the target computer with the storage-manager version.

Importing a Restore

Before you back up the data, set the storage-manager environment variables.

To back up the data and migrate the storage-manager objects

1. Perform a level-0 backup (**onbar -b** or **onbar -b -w**) of all storage spaces on the source database server. (Do not perform an incremental backup.)
2. If you are using ISM, follow these steps:
 - a. Shut down the storage manager on both computers.
 - b. Create a tar file of the storage-manager directories on the source computer.
 - c. Copy this tar file and unpack it on the target computer.

With other storage managers, you might be able to use backup tapes or import the storage-manager directories over the network. For more information, see your storage-manager documentation.

3. Mount the transferred storage volumes.
 - If the backup files are on disk, copy them from the source computer to the target computer.
 - If the backup is on tape, mount the transferred volumes on the storage devices that are attached to the target computer. Both the source and target computers must use the same type of storage devices such as 8mm tape or disk.
 - Some storage managers support remote backups to a backup server. If the backup is on the backup server, retrieve the backup from that backup server.
4. Use storage-manager commands (such as **nsradmin -c**) to add the source hostname as a client on the target computer.

To perform the imported restore

1. Copy the following files from the source computer to the target computer:
 - a. Emergency boot file
 Rename the emergency boot file with the target database server number. For example, rename **ixbar.51** to **ixbar.52**. The emergency boot file needs only the entries from the level-0 backup on the source computer.

 On Dynamic Server, the filename is **ixbar.servernum**. On Extended Parallel Server, the filename is **bixbar.servernum.coservernum**.
 - b. The **oncfg** files: **oncfg_servername.servernum**
 ON-Bar needs the **oncfg** file to know what dbspaces to retrieve. Rename the **oncfg** file with the target database server name and number. For example, rename **oncfg_bostonserver.51** to **oncfg_chicagoserver.52**. The filename should match the DBSERVERNAME and SERVERNUM on the target computer.
 - c. The ONCONFIG file
 In the ONCONFIG file, update the DBSERVERNAME and SERVERNUM parameters with the target database server name and number.
 - d. Storage-manager configuration files, if any
 The storage-manager configuration files might need updating.
2. Use the **onbar -r** command to restore the data.
 If you are importing a whole-system backup, you can use the **onbar -r -w -p** command to restore the data. ♦

Important: Every chunk (including mirrors) must match exactly in size, location, and offset on the source and target computers for the imported restore to complete.

IDS





Initializing High-Availability Data Replication with ON-Bar

Follow the steps for the imported restore first and then start HDR and perform a physical-only restore on the target computer. Also see [“Initializing HDR with an External Backup and Restore”](#) on page 7-23.

Important: If you use ON-Bar to perform the backup and restore, **ontape** is required on both database servers. You cannot remove **ontape** from database servers participating in HDR.

To perform the imported restore

1. Follow the steps in [“To set up an imported restore”](#) on page 6-33.
2. On the source computer, add entries into your **sqlhosts** file or registry to recognize the target instance.

Verify that the source and target database servers can communicate over the network. For more information on **sqlhosts**, see the *Administrator's Guide*.

3. Follow the steps in [“To back up the data and migrate the storage-manager objects”](#) on page 6-34.
4. Copy the emergency boot files, **oncfg** files, ONCONFIG file, and storage-manager files from the source computer to the target computer.

To initialize High-Availability Data Replication

1. To start HDR on the source database server, use the following command:

```
onmode -d primary secondary_dbservername
```

You might see the following messages in the database server message log:

```
19:28:15 DR: new type = primary, secondary server name = solo_724
19:28:15 DR: Trying to connect to secondary server ...
19:28:18 DR: Primary server connected
19:28:18 DR: Receive error
19:28:18 DR: Failure recovery error (2)
19:28:19 DR: Turned off on primary server
19:28:20 Checkpoint Completed: duration was 0 seconds.
19:28:20 DR: Cannot connect to secondary server
19:28:31 DR: Primary server connected
19:28:31 DR: Receive error
19:28:31 DR: Failure recovery error (2)
19:28:32 DR: Turned off on primary server
19:28:33 Checkpoint Completed: duration was 0 seconds.
19:28:33 DR: Cannot connect to secondary server
```

2. Perform a physical-only restore on the target computer.

```
onbar -r -p
```

If you performed a whole-system backup (**onbar -b -w**), you could optionally use **onbar -r -w -p** to restore the storage spaces only.

3. Check the database server message log, ON-Bar activity log, and the storage-manager error log to see whether the restore was successful.
4. To start HDR on the target database server, use the following command:

```
onmode -d secondary primary_dbservername
```

5. If the logical logs needed to synchronize the two database servers are still present on the source database server, the target server retrieves them from the source database server.
6. While the database servers are synchronizing, the logical logs are transferred automatically from the source to the target server.
7. If the logical logs are not on the source database server, you are prompted to restore the required logical logs. If the target database server requires a log number that no longer exists because it was overwritten, ON-Bar will need to retrieve that logical log from the backup.

The following **online.log** messages might display while the database servers are synchronizing:

```
19:37:10 DR: Server type incompatible
19:37:23 DR: Server type incompatible
19:37:31 DR: new type = secondary, primary server name =
bostonserver
19:37:31 DR: Trying to connect to primary server ...
19:37:36 DR: Secondary server connected
19:37:36 DR: Failure recovery from disk in progress ...
19:37:37 Logical Recovery Started.
19:37:37 Start Logical Recovery - Start Log 11, End Log ?
19:37:37 Starting Log Position - 11 0x629c
19:37:44 Checkpoint Completed: duration was 0 seconds.
19:37:45 Checkpoint Completed: duration was 0 seconds.

19:37:47 Checkpoint Completed: duration was 0 seconds.
19:37:48 DR: Secondary server operational
19:37:49 Checkpoint Completed: duration was 0 seconds.
```

Restoring Table Types

[Figure 6-1](#) discusses restore scenarios for different table types. For more information about the table types, see the *Administrator's Guide*.

Figure 6-1
Restoring Table Types

Table Type	IDS	XPS	Can You Restore This Type of Table?
Standard	✓	✓	Yes. Warm restore, cold restore, and point-in-time restore work.
Temp	✓	✓	No.
Scratch		✓	No.

(1 of 2)

Table Type	IDS	XPS	Can You Restore This Type of Table?
Operational		✓	<p>You can restore an operational table if no light appends occurred since the last level-0 backup.</p> <p>If light appends occur to the table since the last backup, the table is not wholly restorable. This sort of problem can also occur if you restore from an older backup. To determine whether your table was restored to its original state, check the message log for the following error message:</p> <pre>Portions of partition partnum of table tablename in database dbname were not logged. This partition cannot be rolled forward.</pre> <p>If you see this message, the table or table fragments were not restored to their original state. If you want full access to whatever remains in this table, you need to alter the table to raw and then to the desired table type. This alter operation removes inconsistencies in the table that resulted from replaying non-logged operations such as light appends.</p>
Raw	✓	✓	<p>When you restore a raw table, it contains only data that was on disk at the time of the last backup. Because raw tables are not logged, changes that occurred since the last backup cannot be restored.</p>
Static		✓	<p>Yes, you can restore the data present at the last dbslice or dbspace backup. Static tables cannot change data. If you alter a static table to another type and update the data, the recoverability of the table depends on each type the table has been since each dbspace was backed up. For example, if you alter a static table to raw, it follows the rules for restoring a raw table.</p>

(2 of 2)

IDS



Using Restartable Restore to Recover Data

If a failure occurs with the database server, media, or ON-Bar during a restore, you can restart the restore from the place that it failed. By default, the `RESTARTABLE_RESTORE` parameter is ON. If it is OFF, you must shut down and restart the database server before you begin the original restore. To restart a failed warm or cold restore, issue the **onbar -RESTART** command. All restarted restores resume where the last restore failed.

Important: Set `RESTARTABLE_RESTORE` to ON if your system is large or unstable. If your system is small, consider turning off restartable restore for faster restore performance only if you have the time to repeat a failed restore from the beginning.



If the failure occurred during a physical restore, ON-Bar restarts the restore at the storage space and level where the failure occurred. It does not matter whether the restore was warm or cold.

If a failure occurred during a cold logical restore, ON-Bar restarts the logical restore from the most recent log checkpoint. Restartable logical restore is supported for cold restores only. However, if the failure during a warm restore caused the database server to shut down, do *not* restart the restore. Instead, use the **archecker** utility to verify the backup and start the whole restore from the beginning.

Warning: Restartable restore does not work for the logical part of a warm restore.

Restartable Restore Example

The following example shows how to use restartable restore for a cold restore:

1. Make sure that `RESTARTABLE_RESTORE` is ON.
If you just set `RESTARTABLE_RESTORE` to ON, shut down and restart the database server for the changes to take effect.

2. Restore several storage spaces:

```
onbar -r rootdbs dbs1 dbs2 dbs3 dbs4
```

The database server fails while restoring **dbs3**.

3. Restart the restore:

```
onbar -RESTART
```

ON-Bar automatically starts restoring **dbs3**, **dbs4**, and the logical logs.

4. If necessary, bring the database server online:

```
onmode -m
```



Important: If a restore fails with `RESTARTABLE_RESTORE` set to OFF, the **onbar -RESTART** option will not work. Use the **onbar -r** command to repeat the restore from the beginning.

Restarting a Restore

You can restart a point-in-time, whole-system, or parallel restore. The physical restore restarts at the storage space and level where the failure occurred. If the restore failed while some, but not all, chunks of a storage space were restored, even a restarted restore must restore all chunks of that storage space again. If storage spaces and incremental backups are restored successfully before the failure, they are not restored again.

Figure 6-2 shows how a restartable restore works when the restore failed during a physical restore of **dbspace2**. For example, you set `RESTARTABLE_RESTORE` to `ON` before you begin the restore. The level-0, level-1, and level-2 backups of **rootdbs**, and the level-0 and level-1 backups of **dbspace1** and **dbspace2** are successfully restored. The database server fails while restoring the level-1 backup of **dbspace2**. When you restart the restore, ON-Bar restores the level-2 backup of **dbspace 1**, the level-1 and level-2 backups of **dbspace2**, and the logical logs.

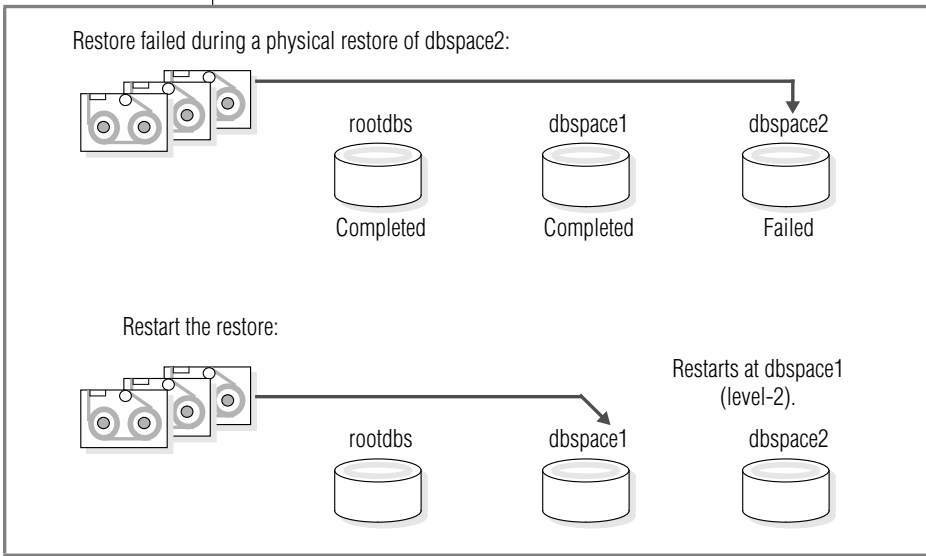


Figure 6-2
Restartable Physical Restore

Interaction Between Restartable Restore and BAR_RETRY Value

If BAR_RETRY > 1, ON-Bar automatically retries the failed storage space or logical log. If this retry is successful, the restore continues and no restart is needed.

If BAR_RETRY = 0 or 1, ON-Bar does not retry the failed storage space or logical log. If the database server is still running, ON-Bar skips the failed storage space and attempts to restore the remaining storage spaces.

Figure 6-3 shows what to expect with different values for BAR_RETRY in a different restarted restore example.

Figure 6-3
Restartable Restore Results with Different BAR_RETRY Values

ON-Bar Command	BAR_RETRY = 2	BAR_RETRY = 0
onbar -r dbs1 dbs2 dbs3	restore level-0 dbs1, dbs2, dbs3 restore level-1 dbs1 FAILS restore level-1 dbs1 RETRY PASSES restore level-1 dbs2, dbs3 restore level-2 dbs1, dbs2, dbs3 restore logical logs	restore level-0 dbs1, dbs2, dbs3 restore level-1 dbs1 FAILS
onbar -RESTART	No restart is needed because everything was successfully restored.	restore level-1 dbs1, dbs2, dbs3 restore level-2 dbs1, dbs2, dbs3 restore logical logs
onbar -r dbs1 dbs2 dbs3	restore level-0 dbs1, dbs2, dbs3 restore level-1 dbs1 FAILS restore level-1 dbs1 RETRY FAILS restore level-1 dbs2, dbs3 restore level-2 dbs2, dbs3 restore logical logs	restore level-0 dbs1, dbs2, dbs3 restore level-1 dbs1 FAILS
	onbar -r dbs1 dbs2 restore level-1 dbs1 restore level-2 dbs1 restore logical logs	onbar -RESTART restore level-1 dbs1, dbs2, dbs3 restore level-2 dbs1, dbs2, dbs3 restore logical logs

Restarting a Logical Restore

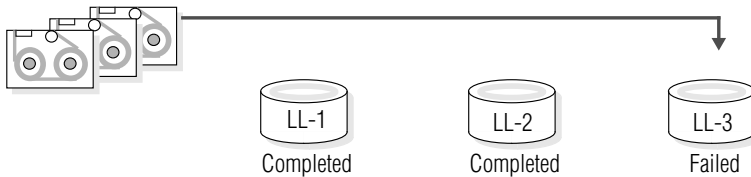
If a restore fails during the logical phase and you restart the restore, ON-Bar verifies that the storage spaces have been restored successfully, skips the physical restore, and restarts the logical restore. [Figure 6-4 on page 6-43](#) shows a cold restore that failed while restoring logical log LL-3. When you restart the cold logical restore, log replay starts from the last restored checkpoint. In this example, the last checkpoint is in logical log LL-2.

If a failure occurs during a cold logical restore, ON-Bar restarts it at the place that it failed.



Important: If a failure occurs during a warm logical restore, you have to restart it from the beginning. If the database server is still running, use the **onbar -r -l** command to complete the restore.

Cold restore failed during a logical restore of LL-3. The last checkpoint is in LL-2.



Restart the cold restore:



Figure 6-4
Restartable Cold
Logical Restore

Set the `RESTARTABLE_RESTORE` parameter to `ON`. A restartable restore makes the logical restore run more slowly if many logical logs need to be restored, but it saves you time if something goes wrong and you need to restart. Restartable restore does not affect the speed of the physical restore.

Resolving a Failed Restore

What is retried, what is restartable, and what command you use to restart the restore depends on what failed and how serious it was. You can save some failed restores even if restartable restore is turned off. For example, if the restore fails because of a storage-manager or storage-device error, you can fix the tape drive or storage-manager problem, remount a tape, and then continue the restore.

Figure 6-5 shows what results to expect when physical restore fails. Assume that `BAR_RETRY > 1` in each case.

Figure 6-5
Failed Physical Restore Scenarios

Type of Error	RESTARTABLE_RESTORE	What to Do When the Physical Restore Fails?
Database server, ON-Bar, or storage-manager error (database server is still running)	ON or OFF	ON-Bar retries each failed restore. If the storage manager failed, fix the storage-manager error. If the retried restore fails, issue onbar -r spaces where <i>spaces</i> is the list of storage spaces not yet restored. Use onstat -d to obtain the list of storage spaces that need to be restored. ON-Bar restores the level-0 backup of each storage space, then the level-1 and level-2 backups, if any.
ON-Bar or storage-manager error (database server is still running)	ON	Issue the onbar -RESTART command. If the storage manager failed, fix the storage-manager error. The restore restarts at the storage space and backup level where the first restore failed. If the level-0 backup of a storage space was successfully restored, the restarted restore skips the level-0 backup and restores the level-1 and level-2 backups, if any.
Database server failure	ON or OFF	Because the database server is down, perform a cold restore. Use onbar -r to restore the critical dbspaces and any noncritical spaces that were not restored the first time.
Database server failure	ON	Issue the onbar -RESTART command. The restore restarts at the storage space and backup level where the first restore failed. If the level-0 backup of a storage space was successfully restored, the restarted restore skips the level-0 backup and restores the level-1 and level-2 backups, if any.

Figure 6-6 shows what results to expect when logical restore fails.

Figure 6-6
Failed Logical Restore Scenarios

Type of Error	RESTARTABLE_RESTORE	What to Do When a Logical Restore Fails?
Database server or ON-Bar error in a cold restore (database server is still running)	ON	Issue the onbar -RESTART command. The logical restore restarts at the last checkpoint. If this restore fails, shut down and restart the database server to initiate fast recovery of the logical logs. All logical logs not restored are lost.
Database server or ON-Bar error (database server is still running)	ON or OFF	Issue the onbar -r -l command. The restore should restart at the failed logical log. If onbar -r -l still fails, shut down and restart the database server. The database server will complete fast recovery. All logical logs that were not restored are lost. If fast recovery does not work, you have to do a cold restore.
Database server error	ON	If the cold logical restore failed, issue onbar -RESTART . If the warm logical restore failed, issue the onbar -r -l command. If that fails, restart the entire restore from the beginning.
Storage-manager error (IDS)	ON or OFF	ON-Bar retries each failed logical restore. If the retried restore fails, the logical restore is suspended. Fix the storage-manager error. Then issue the onbar -r -l command. The restore should restart at the failed logical log.

Understanding ON-Bar Restore Processes

This section explains how ON-Bar performs restore operations on the database server. If the database server is in quiescent mode or is online, you can perform a warm restore. ON-Bar gathers storage-space and logical-log backup data from the **sysutils** database and then requests a restore from the database server.

If you have lost critical dbspaces, you must perform a cold restore. ON-Bar gathers backup data from the emergency boot file and then restores the storage spaces and logical logs.

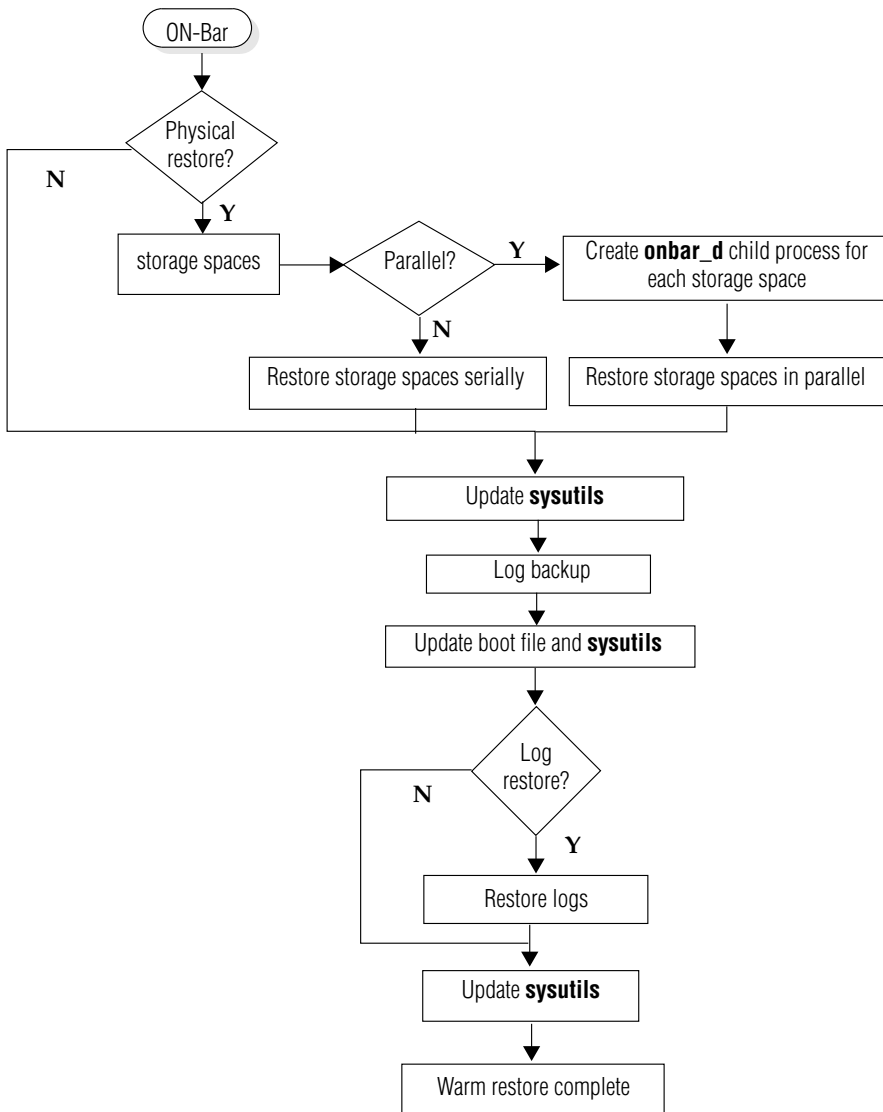
Warm-Restore Sequence on Dynamic Server

[Figure 6-7 on page 6-47](#) describes the ON-Bar warm-restore sequence.

In a warm restore, the **onbar-driver** creates a list of restore objects. In a parallel restore (if **BAR_MAX_BACKUP** is not set to 1), the ON-Bar driver starts **onbar_d** child processes. The **onbar_d** processes transfer data between the storage manager and the database server until the warm restore is complete. Each **onbar_d** process processes one storage space. In a serial restore, the **onbar-driver** restores the storage spaces one at a time. Then the **onbar-driver** performs the logical backup and restore. After each object is restored, information about it is added to the **sysutils** database.

For each storage space, ON-Bar restores the last level-0 backup, the level-1 backup (if it exists), and the level-2 backup (if it exists). After the physical restore is complete, ON-Bar backs up the logical logs to get the latest checkpoint and then restores the logical logs. This logical backup allows data to be restored up to the moment of failure.

Figure 6-7
ON-Bar Warm-Restore Sequence on Dynamic Server



Cold-Restore Sequence on Dynamic Server

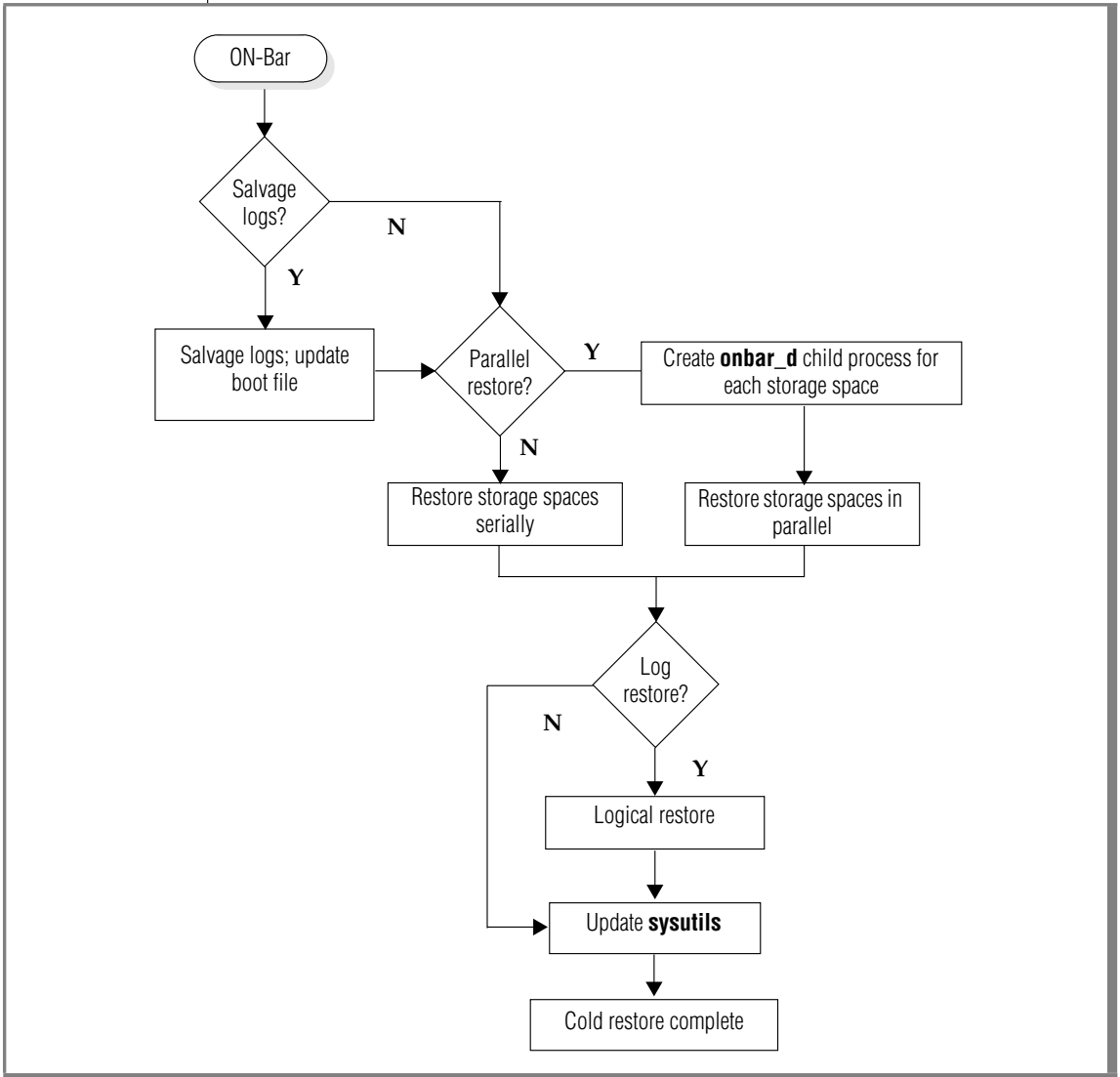
[Figure 6-8 on page 6-49](#) describes the ON-Bar cold-restore sequence. ON-Bar uses the backup emergency boot file to determine what restores are required.

In a cold restore, ON-Bar performs the following steps in order:

- Salvages the logical logs
- Restores the root dbspace
- Restores the critical dbspaces
- Restores blobspaces
- Restores noncritical dbspaces and sbspaces
- Restores logical logs

For each storage space, ON-Bar restores the last level-0 backup, the level-1 backup (if it exists), and the level-2 backup (if it exists). Finally, ON-Bar restores the logical logs.

Figure 6-8
ON-Bar Cold-Restore Sequence on Dynamic Server



Warm-Restore Sequence on Extended Parallel Server

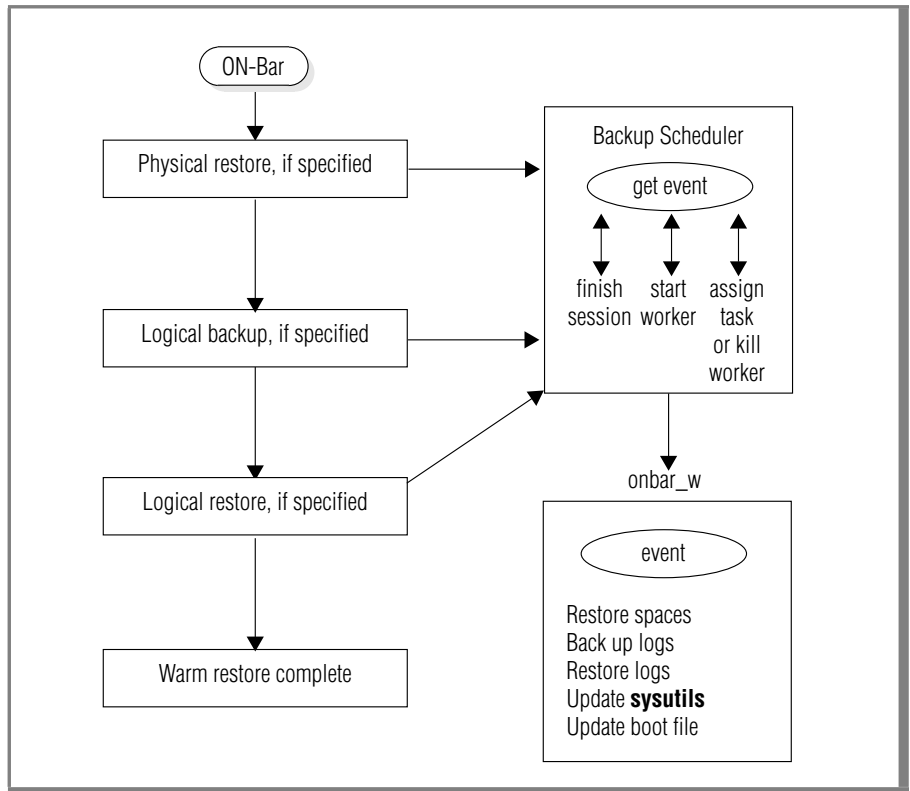
[Figure 6-9 on page 6-51](#) describes the ON-Bar warm-restore sequence.

In a warm restore, the **onbar-driver** sends a list of backup objects to the Backup Scheduler. The Backup Scheduler creates a restore session that contains lists of backup objects to restore and might start one or more **onbar-worker** processes. The **onbar-worker** transfers data between the storage manager and the database server until the warm restore is complete. For each storage space, ON-Bar restores the last level-0 backup, the level-1 backup (if it exists), and the level-2 backup (if it exists). Next, ON-Bar backs up the logical logs to get the latest checkpoint and then restores them. This logical-log backup allows data to be restored as close to the moment of failure as possible.

As each object is restored, information about the restore is added to the **sysutils** database. As each logical log is backed up, information about it is added to **sysutils** and the emergency boot file. The emergency backup boot file is on the coserver of the **onbar-worker** that backed it up.

Figure 6-9

ON-Bar Warm-Restore Sequence on Extended Parallel Server



Cold-Restore Sequence on Extended Parallel Server

You must put the database server in microkernel mode to do a cold restore. If the database server or a coserver is offline, you cannot perform any restores. [Figure 6-10 on page 6-53](#) describes the ON-Bar cold-restore sequence.

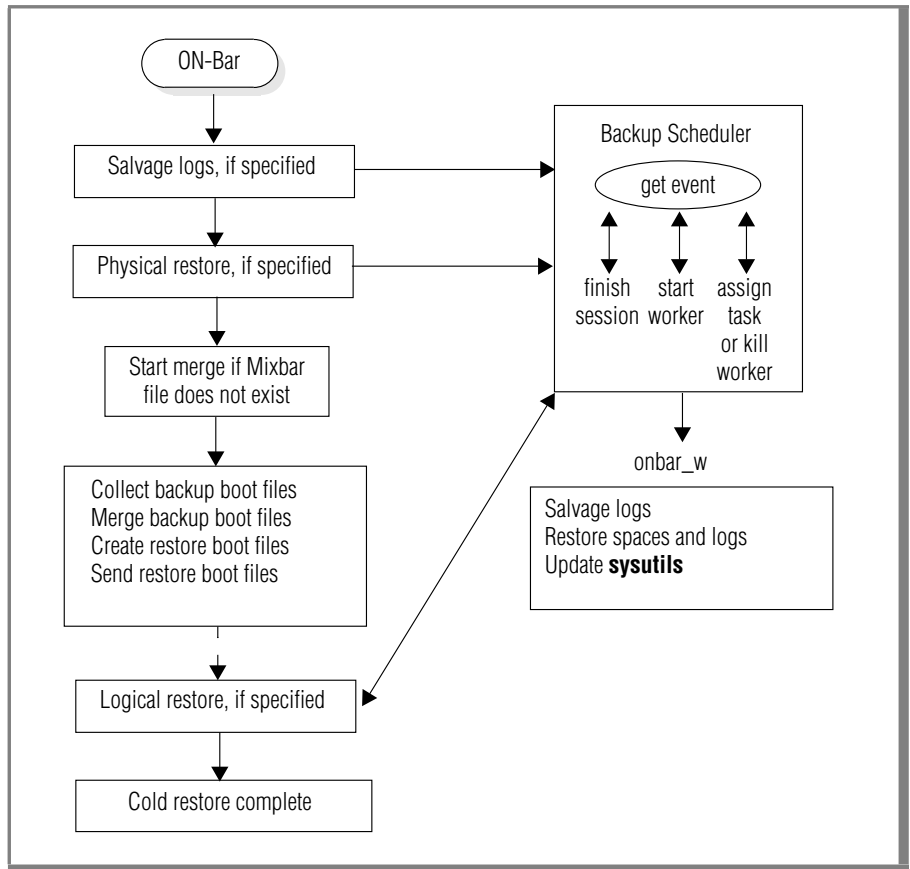
In a cold restore, ON-Bar performs the following steps in order:

- Salvages the logical logs
- Merges the boot files
- Restores the root dbspace
- Restores the critical dbspaces
- Restores the other dbspaces
- Restores logical logs

The **onbar-merger** utility collects and processes the backup emergency boot files from each coserver to determine what restores are required. The **onbar-merger** then creates the restore boot file and copies it to each coserver that contains a backup emergency boot file.

You can specify in the `BAR_WORKER_COSERVER` configuration parameter which coservers have boot files and run **onbar-worker** processes. For more information, see [“BAR_WORKER_COSVR” on page 9-20](#).

For each storage space, ON-Bar restores the last level-0 backup, the level-1 backup (if it exists), and the level-2 backup (if it exists). Finally, it restores the logical logs.

Figure 6-10*ON-Bar Cold-Restore Sequence on Extended Parallel Server*

Performing an External Backup and Restore

In This Chapter	7-3
Recovering Data Using an External Backup and Restore	7-3
What Is Backed Up in an External Backup?	7-5
Rules for an External Backup	7-5
Performing an External Backup	7-7
Preparing for an External Backup	7-9
Blocking and Unblocking Dynamic Server	7-9
Blocking and Unblocking Extended Parallel Server	7-10
Blocking Coservers or the Database Server	7-10
Blocking a Cogroup and Specifying a Session Name.	7-11
Blocking Coservers	7-11
Unblocking Coservers or the Database Server	7-12
Unblocking All Coservers and Marking Dbspaces as Backed Up	7-14
Unblocking a Cogroup and Marking Selected Dbspaces as Backed Up	7-14
Unblocking a Specific Session and Not Marking Dbspaces as Backed Up	7-14
Monitoring an External Backup	7-15
Tracking an External Backup	7-15
What Is Restored in an External Restore?	7-16
Using External Restore Commands	7-17
Rules for an External Restore	7-18
Performing an External Restore	7-19
Cold External Restore Procedure	7-19
Mixed External Restore Restriction.	7-20

Warm External Restore Procedure	7-21
Examples of External Restore Commands	7-22
Initializing HDR with an External Backup and Restore	7-23

In This Chapter

This chapter discusses recovering data using external backup and restore.

Recovering Data Using an External Backup and Restore

An external backup and restore eliminates the downtime of systems because the backup and restore operations are performed external to the Informix system. ON-Bar does not move the data during the backup or physical restore. An external backup allows you to copy disks that contain storage-space chunks without using ON-Bar. When disks fail, replace them and use third-party software to restore the data, then use ON-Bar for the logical restore. For more information, see [“What Is Restored in an External Restore?” on page 7-16](#).

The following are typical scenarios for external backup and restore:

- *Availability with disk mirroring.* If you use hardware disk mirroring, you can get your system online faster with external backup and restore than with conventional ON-Bar commands.
- *Cloning.* You can use external backup and restore to clone an existing production system for testing or migration without disturbing the production system.

Figure 7-1 shows how an external backup and restore moves the data directly between the storage spaces on disk and the storage media. You can externally back up and restore specific storage spaces or all storage spaces but not logical logs.

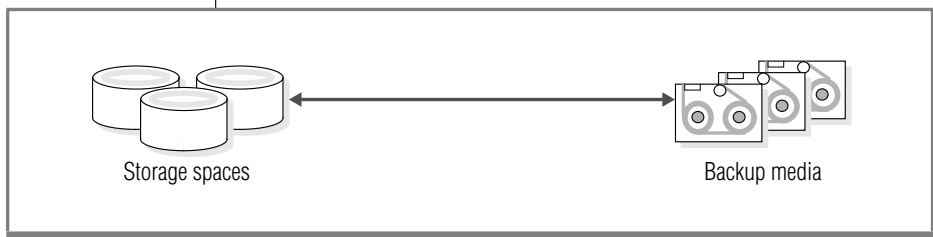


Figure 7-1
Step 1: Moving the Data in an External Backup and Restore

Figure 7-2 shows how an external restore uses the **onbar -r -e** command to restore the logical logs.

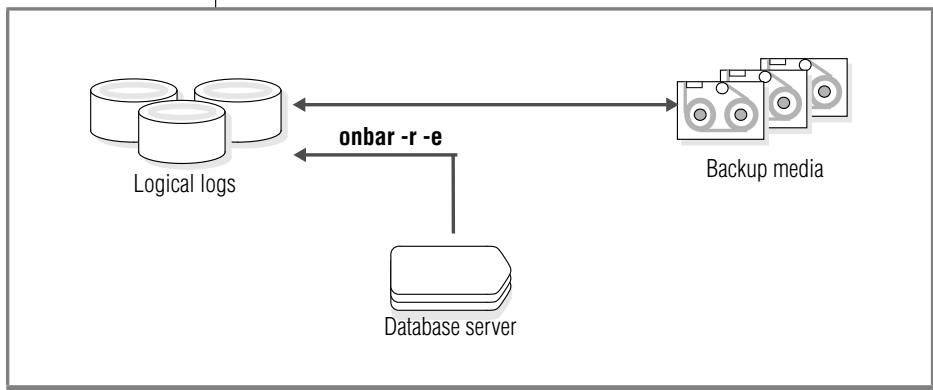


Figure 7-2
Step 2: Restoring the Logical Logs in an External Restore

What Is Backed Up in an External Backup?

Before you begin an external backup, block the database server or one or more coservers. Blocking forces a checkpoint, flushes buffers to disk, and blocks user transactions that involve temporary tables. During the blocking operation, users can access that database server or coserver in read-only mode. Then you can physically back up or copy the data to another set of disks or storage media using operating-system or third-party tools. When you complete the external backup, unblock the database server so that transactions can resume. You should include all the chunk files in each storage space, administrative files, such as **ONCONFIG**, and the emergency boot file, in an external backup.



Important: *To make tracking backups easier, it is recommended that you back up all storage spaces in each external backup.*

ON-Bar treats an external backup as equivalent to a level-0 backup. You cannot perform an external backup and then use ON-Bar to perform a level-1 backup, or vice versa because ON-Bar does not have any record of the external backup. For more information, see [“Performing an External Backup” on page 7-7](#).

Rules for an External Backup

Before you begin an external backup, keep the following rules in mind:

- The database server must be online or quiescent during an external backup.
- Use ON-Bar to back up all logical logs including the current log so that you can restore the logical logs at the end of the external restore.

XPS

IDS



- Suspend continuous logical-log backups before you block the database server for an external backup. After the external backup is complete, resume the continuous logical-log backup.

To suspend continuous logical-log backup, use the **onbar off Log backup coserver_id** command. To resume continuous logical-log backup, use the **onbar on Log backup coserver_id** command. For more information, see [“Starting and Stopping ON-Bar Sessions” on page 8-14](#). ♦

To stop continuous logical-log backup, use the CTRL-C command. To resume continuous logical-log backup, use the **onbar -b -l -C** command. ♦

- Wait until all ON-Bar backup sessions have completed before you block the database server. If any backup sessions are active, the block command displays an error message.
- Any OLTP work or queries are suspended while the database server is blocked. They resume after the database server is unblocked.
- All critical dbspaces of the database server instance must be backed up together simultaneously within the same command bracket of **onmode -c block ... onmode -c unblock**. Backups of different critical dbspaces done at different times cannot be restored to a consistent system.

Important: Because the external backup is outside the control of ON-Bar, you must track these backups manually. For more information, see [“Tracking an External Backup” on page 7-15](#).

Performing an External Backup

The database server must be online or in quiescent mode during an external backup.

To perform an external backup without disk mirroring

1. To obtain an external backup, block the database server. The system takes a checkpoint and suspends all update transactions. Users can access the database server in read-only mode.

On Dynamic Server, use the following command:

```
onmode -c block
```



On Extended Parallel Server, use the following command to block all the coservers in **cogroup_all**:

```
onutil ebr block;
```



2. To back up the storage spaces and administrative files, use a copy command, such as **cp**, **dd**, or **tar** on UNIX or **copy** on Windows, or a file-backup program. You must back up all chunks in the storage spaces.
3. To allow normal operations to resume, unblock the database server.

Use the following command:

```
onmode -c unblock
```



Use the following command to unblock all the coservers in **cogroup_all** and mark all dbspaces on each coserver as having a level-0 backup:

```
onutil ebr unblock commit;
```



4. Back up all the logical logs including the current log so that checkpoint information is available for the external restore.

Warning: Because external backup is not done through ON-Bar, you must ensure that you have a backup of the current logical log from the time when you execute the **onutil EBR BLOCK** or **onmode -c block** command. Without a backup of this logical-log file, the external backup is not restorable.

IDS

XPS

IDS

XPS



IDS

5. After you perform an external backup, use the following command to back up the current log on Dynamic Server:

```
onbar -b -l -c
```



After you perform an external backup, use the following commands to switch to and back up the current log on Extended Parallel Server:

```
onmode -l # execute on coservers with external backups  
onbar -b -l # back up all used logs
```



If you lose a disk, coserver, or the whole system, you are now ready to perform an external restore.

To perform an external backup with hardware disk mirroring

For example, you could set up a high-availability environment and use external backups for fail-over to a second database server.

1. If you are using hardware to mirror your disks, copy the primary partition to the mirror partition.
2. Synchronize the mirrored disk partitions.
3. Block the database server. The system takes a checkpoint and suspends all update transactions.
4. Break the link between the disk mirrors. Install a new set of disk mirrors, if needed.
5. Unblock the database server so that transactions can resume.
6. Back up the logical logs.

On Dynamic Server only, also back up the current log. ◆

7. Put the mirror disk into storage, copy the mirrored data to another computer, or back it up to tape or storage media by using a file-system backup program or an operating-system copy command.
8. Reconnect and synchronize the disk mirrors, if necessary.

IDS

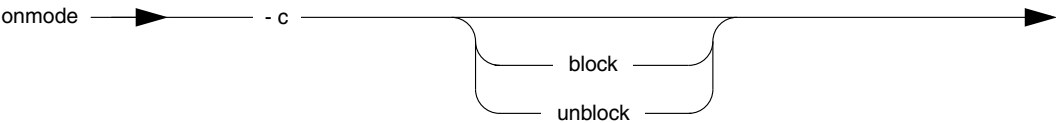
Preparing for an External Backup

This section describes the commands used to prepare for an external backup. For the procedure, see [“Performing an External Backup” on page 7-7](#).

IDS

Blocking and Unblocking Dynamic Server

This section shows the syntax of the block and unblock commands on Dynamic Server.



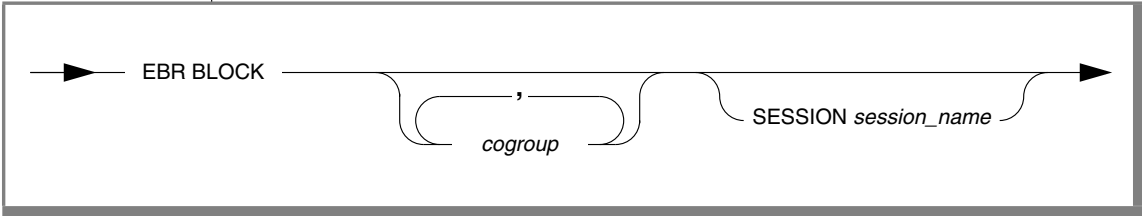
Element	Purpose	Key Considerations
-c	Performs a checkpoint and blocks or unblocks the database server	None.
block	Blocks the database server from any transactions	Sets up the database server for an external backup. While the database server is blocked, users can access it in read-only mode. Sample command: onmode -c block
unblock	Unblocks the database server, allowing data transactions and normal database server operations to resume	Do not unblock until the external backup is finished. Sample command: onmode -c unblock

Blocking and Unblocking Extended Parallel Server

This section shows the syntax of the block and unblock commands on Extended Parallel Server.

Blocking Coservers or the Database Server

Use the **onutil EBR BLOCK** command to block the database server or coservers. The EBR, BLOCK, and SESSION keywords can be uppercase or lowercase. Before you begin an external backup, block the database server or coserver. You can leave the coservers blocked for as long as you decide.



Element	Purpose	Key Considerations
BLOCK	Forces a checkpoint that flushes the buffers to disk and blocks the database server or specified coservers from any transactions	While a coserver or database server is blocked, users can access it in read-only mode.
cogroup	Specifies the set of coservers to block You can block one or more listed coservers or cogroups, or cogroup_all . If you specify coserver names, use the format: <i>dbservername.coserverid</i> . If you specify a cogroup range identifier, use the format: <i>dbservername.%r(first.last)</i> .	If no cogroup or coserver name is specified, all coservers on the database server are blocked.
EBR	Specifies an external backup or restore command	None.
SESSION session_name	Assigns a session name to the onutil EBR BLOCK command Use the session name to monitor the external backup status.	If no session name is specified, onutil generates a session name of the following format: ebr_connection-coserver-id.client-sql-session-id.ebr-command-number .

When a coserver is blocked, all the data on disk is synchronized and users can access that coserver in read-only mode. When you block a coserver, the entire coserver is blocked. You can block a list of coservers, cogroups, or the entire database server. You can assign a session name to the blocking operation.

To block the entire database server (all the coservers in **cogroup_all**), use either of the following commands:

```
onutil ebr block;
```

or

```
onutil ebr block cogroup_all;
```

When the block command completes, the default session ID is printed to standard output and a message is written to **online.log**.

After you complete the external backup, unblock the database server. For the syntax diagram, see [“Unblocking Coservers or the Database Server” on page 7-12](#).

Blocking a Cogroup and Specifying a Session Name

Specifying a session name is useful for monitoring external backup sessions. To block the coservers defined in cogroup **data_cogroup** and to name the Backup Scheduler for that cogroup as **block_data_cogroup**, use the following command:

```
onutil ebr block data_cogroup session block_data_cogroup;
```

For information on how to create cogroups, see the **onutil** section of the *Administrator's Reference*. For information on how to monitor this session, see [“Monitoring an External Backup” on page 7-15](#).

Blocking Coservers

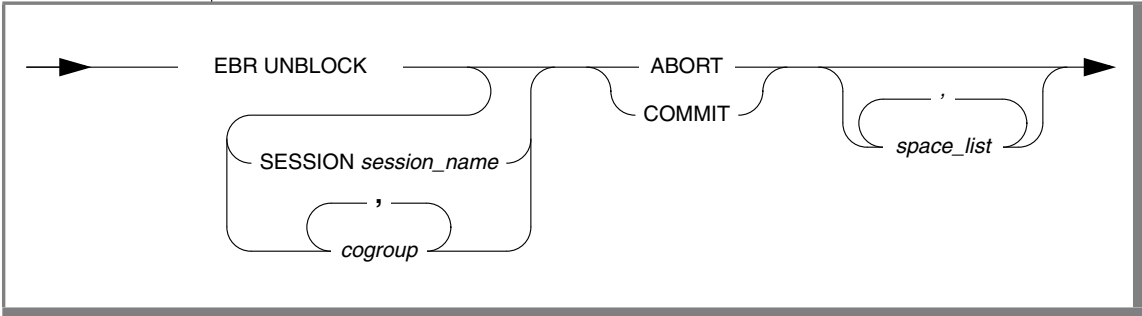
To block specific coservers, either specify them by name or use the cogroup range identifier with the **onutil** command. For example, **xps.%r(1.4)** expands to coservers **xps.1**, **xps.2**, **xps.3**, and **xps.4**. The following example blocks coservers **xps.1** and **xps.2**.

```
onutil ebr block xps.1, xps.2;
```

For information on how to specify coserver names and cogroup range identifiers, see the **onutil** section of the *Administrator's Reference*.

Unblocking Coservers or the Database Server

After you complete the external backup, use the **onutil** EBR UNBLOCK command to unblock and allow write access for data transactions. The EBR, UNBLOCK, SESSION, ABORT, and COMMIT keywords can be uppercase or lowercase.



Element	Purpose	Key Considerations
ABORT	Unblocks the specified coservers but does not mark the dbspaces as backed up	Use the ABORT option when the external backup failed or when you do not want to update the backup status. The dbspaces are not restorable.
cogroup	Specifies a list of one or more cogroup names to unblock the coservers You can specify cogroup_all , cogroup names, coserver names, or a cogroup range identifier.	All coservers in the specified cogroups that are currently blocked are unblocked. If you do not specify a cogroup name, all the coservers in the database server are unblocked. If a specific coserver is not blocked, an error is reported.

(1 of 2)

Element	Purpose	Key Considerations
COMMIT	Marks the listed dbspaces or dbslices as successfully backed up while the set of coservers were blocked Enables newly added logical logs and dbspace mirrors. If no logged updates have occurred since the last backup, marks the dbspace for skipping log replay. To verify, issue onstat -d .	If you do not specify any dbspaces or dbslices after the COMMIT keyword, it commits the dbspaces and dbslices on the coservers that were blocked.
EBR	Specifies an external backup or restore	None.
SESSION <i>session_name</i>	Specifies a session name that can be either a name that you specified or onutil automatically created in the block command	The session name must be one that was used in a previous block command. All coservers that were part of that session are now unblocked.
<i>space_list</i>	Specifies the dbspaces or dbslices that are successfully backed up	Separate the dbspace or dbslice names with commas.
UNBLOCK	Unblocks the coservers or database server, allowing data transactions and normal database server operations to resume	Ends the external backup.

(2 of 2)



To unblock a list of coservers or cogroups, or the entire database server, you can specify a session name. The list of coservers in the **onutil** EBR UNBLOCK command does not have to match the list of coservers in the **onutil** EBR BLOCK command.

To mark selected or all dbspaces and dbslices as backed up on each blocked coserver, use the **COMMIT** option. Use the **ABORT** option when you do not want to mark any dbspaces as backed up.

Tip: Be sure not to unblock coservers that another user has blocked. (For information, see *“Monitoring an External Backup” on page 7-15.*)

Unblocking All Coservers and Marking Dbspaces as Backed Up

To unblock the entire database server (all the coservers in **cogroup_all**), use either of the following commands. The **COMMIT** option marks all dbspaces on each coserver as having a level-0 backup.

```
onutil ebr unblock commit;  
  
or  
  
onutil ebr unblock cogroup_all commit;
```

Unblocking a Cogroup and Marking Selected Dbspaces as Backed Up

The following example unblocks the coservers in cogroup **data_group** and marks selected dbspaces or dbslices as backed up. The **COMMIT** option marks dbspaces **rootdbs.1**, **rootdbs.2**, and **rootdbs.3** as backed up.

```
onutil ebr unblock data_group commit rootdbs.%r(1..3);
```

Unblocking a Specific Session and Not Marking Dbspaces as Backed Up

Use the **ABORT** option to cancel a failed external backup and unblock the set of coservers. You cannot externally restore these dbspaces because the backups are incomplete.

If you use the **ABORT** option, the skip logical replay feature does not work, even if that dspace was completely backed up. In this case, use the **COMMIT** option to commit those dbspaces that were backed up successfully and use the **ABORT** option for the remaining dbspaces.

The following example unblocks the coservers identified in session **block_data_cogroup** and does not mark any dbspaces as backed up. You must have named the session in the previous block command. The **ABORT** option means that no dbspaces are marked as backed up.

```
onutil ebr unblock session block_data_cogroup abort;
```

Monitoring an External Backup

To monitor the external backup status, use the **onstat -g -bus** command. To find blocked coservers, use the **xctl onstat -** command. For more information, see [“Using the onstat -g bus Option” on page 8-17](#).

To monitor the external backup status in the **sysmaster** database, use the following SQL queries. For more information, see [“Backup Scheduler SMI Tables” on page 10-11](#).

```
# to find all blocked coservers
SELECT * FROM sysbuobject
WHERE is_block = 1 AND is_coserver = 1;

# to find all blocked coservers in my session
SELECT * FROM sysbuobjses WHERE os_ses_id = "my_session";
```

Tracking an External Backup

The database server and ON-Bar do *not* track external backups. To track the external backup data, use a third-party storage manager or track the data manually. [Figure 7-3](#) shows which items we recommend that you track in an external backup. ON-Bar keeps a limited history of external restores.

Figure 7-3
Items to Track When You Use External Backup and Restore

Items to Track	Examples
Full pathnames of each chunk file for each backed up storage space	/work/dbspaces/rootdbs (UNIX) c:\work\dbspaces\rootdbs (Windows)
Object type	Critical dbspaces, noncritical storage spaces
ins_copyid_hi and ins_copyid_lo	Copy ID that the storage manager assigns to each backup object
Backup date and time	The times that the database server was blocked and unblocked
Backup media	Tape volume number or disk pathname
Database server version	Version 9.4 or Version 8.40

What Is Restored in an External Restore?

If you lose a disk, coserver, or the whole system, you can externally restore data only if it was externally backed up. You must use the same third-party utility for both the external backup and restore. To externally restore the storage spaces, copy the backed up data to disk. Use the **onbar -r -e** command to mark the storage spaces as physically restored, replay the logical logs, and bring the storage spaces back online. If you do not specify an external restore command, the database server thinks that these storage spaces are still down.

You can perform these types of external restores:

- **Warm external restore.** Mark noncritical storage spaces as physically restored, then perform a logical restore of these storage spaces.
- **Cold external restore.** Mark storage spaces as physically restored, then perform a logical restore of all storage spaces. Optionally, you can do a point-in-time cold external restore.



Warning: When you perform a cold external restore, ON-Bar does not first attempt to salvage logical-log files from the database server because the external backup has already copied over the logical-log data.

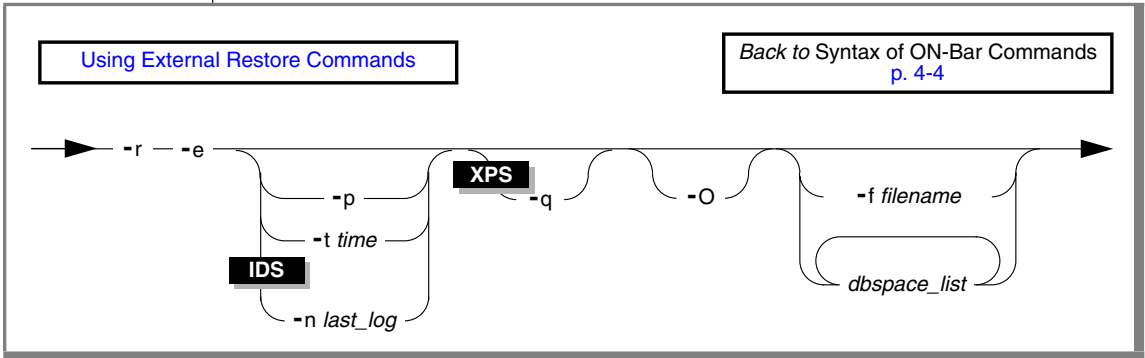
To salvage logical logs, perform **onbar -l -s** before you copy the external backup and perform the external restore (**onbar -r -e**).

XPS

In Extended Parallel Server, you must perform a logical restore on the whole system after an external backup even if all the storage spaces were backed up together. ♦

Using External Restore Commands

Use the **onbar -r -e** command to perform a warm or cold external restore. This command marks the storage spaces as physically restored and restores the logical logs. The following diagram shows the external restore syntax.



Element	Purpose	Key Considerations
-r	Specifies a restore	In a cold restore, if you do not specify storage space names, all of them are marked as restored.
-e	Specifies an external restore	Must be used with the -r option. In a warm external restore, marks the down storage spaces as restored unless the -O option is specified.
dbspace_list	Names one or more storage spaces to be marked as restored in a warm restore	If you do not enter dbspace_list or -f filename and the database server is online or quiescent, ON-Bar marks only the down storage spaces as restored. If you enter more than one storage-space name, use a space to separate the names.
-f filename	Restores the storage spaces that are listed in the text file whose pathname <i>filename</i> provides	To avoid entering a long list of storage spaces every time, use this option. The filename can be any valid UNIX or Windows filename.
-n last_log	Indicates the number of the last log to restore (IDS)	If any logical logs exist after this one, ON-Bar does not restore them and data is lost. The -n option does not work with the -p option.
-O	Restores online storage spaces	None.
-p	Specifies an external physical restore only	After the physical restore completes, you must perform a logical restore because XPS does not support whole-system restore.

(1 of 2)

Element	Purpose	Key Considerations
-q name	Allows you to assign a session name to the external restore (XPS)	<DBSERVERNAME><random_number> is the default session name. The session name must be unique and can be up to 128 characters.
-t time	Restores the last backup before the specified point in <i>time</i> If you pick a backup made after the point in time, the restore will fail.	You can use a point-in-time restore in a cold restore only. You must restore all storage spaces. How you enter the time depends on your current GLS locale convention. If the GLS locale is not set, use English-style date format. See “Restoring Data to a Point in Time” on page 6-22.

(2 of 2)

Rules for an External Restore

Before you begin an external restore, keep the following rules in mind:

- You must externally restore from an external backup. Although the external backup is treated as a level-0 backup, it might actually be a non-Informix incremental backup.
- A warm external restore restores only noncritical storage spaces.
- You cannot externally restore temporary dbspaces.
- You cannot externally restore from regular ON-Bar backups.
- You cannot verify that you are restoring from the correct backup and that the storage media is readable using ON-Bar.
- If the external backups are from different times, the external restore uses the beginning logical log from the oldest backup.

These rules apply to cold external restores only:

- Salvage the logical logs (**onbar -b -l -s**) before you switch the disks that contain the critical storage spaces.
- If you are restoring critical dbspaces, the database server *must* be offline (Dynamic Server) or in microkernel mode (Extended Parallel Server).
- Point-in-time external restores must be cold and restore all storage spaces.
- If you are restoring the rootdbs, disable mirroring during the restore.
- The external backups of all critical dbspaces of the database server instance must have been simultaneous. All critical dbspaces must have been backed up within the same **onmode -c block ... onmode -c unblock** command bracket.

Performing an External Restore

This section describes procedures for performing cold and warm external restores.

Cold External Restore Procedure

If you specify the **onbar -r -e** command in a cold restore, you must restore all storage spaces. Use the **onbar -r -e -p** command to restore all or specific storage spaces.

To perform a cold external restore

1. To shut down the database server, use the **onmode -ky** command. ♦
To bring the database server to microkernel mode, use the **xctl -C oninit -m** command. ♦
2. Salvage the logical logs:

```
onbar -b -l -s
```
3. To restore the storage spaces from an external backup, use a copy command, such as **cp**, **dd**, or **tar** on UNIX or a file-backup program. You must restore the storage spaces to the same path as the original data and include all the chunk files.

IDS

XPS

4. To perform an external restore of all storage spaces and logical logs, use the following command:

```
onbar -r -e
```

To perform a point-in-time external restore of all storage spaces, use the following command:

```
onbar -r -e -t datetime
```

To perform a physical external restore of specific storage spaces, followed by a logical restore, use these commands:

```
onbar -r -e -p rootdbs
onbar -r -e -p critical_space1
onbar -r -e -p other_dbspaces
onbar -r -l
```

This step brings the database server to fast-recovery mode.

5. ON-Bar and the database server roll forward the logical logs and bring the storage spaces online.

Mixed External Restore Restriction

ON-Bar does not support mixed external restores. For example, the following sequence of commands might fail:

```
onbar -r -e rootdbs
onbar -r -e other_dbspaces
```


Warm External Restore Procedure

The database server is online during a warm external restore. A warm external restore involves only noncritical storage spaces.

To perform a warm external restore

1. To restore the storage spaces from an external backup, use a copy command, such as **cp**, **dd**, or **tar** on UNIX or a file-backup program. You must restore the storage spaces to the same path as the original data and include all the chunk files for each restored storage space.
2. Perform a warm external restore of the noncritical storage spaces to bring them online.

- a. To restore selected storage spaces and all logical logs, use the following command:

```
onbar -r -e dbspace_list
```

- b. To restore all the down noncritical storage spaces and logical logs, use the following command:

```
onbar -r -e
```

- c. To restore all the down noncritical storage spaces and logical logs in separate steps, use the following commands:

```
onbar -r -e -p
onbar -r -l
```

Specify the logstreams for only the coservers from where you restored the dbspaces. For example, use the following command to restore logstreams 1, 2, and 3:

```
onbar -r -l 1 2 3
```

- d. To restore *all* the noncritical storage spaces and logical logs, use the following command:

```
onbar -r -e -O
```

Examples of External Restore Commands

The following table contains examples of external restore commands.

Action	External Restore Command	Comments
Complete external restore	onbar -r -e	In a cold restore, restores everything. In a warm restore, restores all down noncritical storage spaces.
Physical external restore and separate logical restore	onbar -r -e -p onbar -r -l	You must always perform a logical restore (XPS). If the external backups come from different times, you must perform a logical restore. The system restores the logical logs from the oldest external backup (IDS).
External restore of selected storage spaces and logical logs	onbar -r -e <i>dbspace_list</i>	Use this command in a warm external restore only.
External restore of selected storage spaces and separate logical restore	onbar -r -e -p <i>dbspace_list</i> onbar -r -l	Use this command in a warm or cold external restore.
External point-in-time (cold) restore	onbar -r -e -t <i>datetime</i>	Be sure to select a collection of backups from before the specified time.
Whole-system external restore (IDS)	onbar -r -e -w or onbar -r -e -p -w	When you use onbar -r -e -w -p , back up all storage spaces in one block and unblock session. That way, all storage spaces have the same checkpoint.

Initializing HDR with an External Backup and Restore

You can use external backups to initialize High-Availability Data Replication (HDR). For more information on HDR, see [“Initializing High-Availability Data Replication with ON-Bar”](#) on page 6-36 and the *Administrator’s Guide*.

To initialize HDR with an external backup and restore

1. Block the source database server with the following command:

```
onmode -c block
```
2. Externally back up all chunks on the source database server.
3. When the backup completes, unblock the source database server:

```
onmode -c unblock
```
4. Make the source database server the primary server:

```
onmode -d primary secondary_servername
```
5. On the target database server, restore the external backup of all chunks.
6. Bring the target database server to recovery mode:

```
oninit -r
```
7. Make the target database server the secondary server:

```
onmode -d secondary primary_servername
```
8. Wait until the “database server operational” messages appear in the message log on the primary and secondary servers.

Using ONBar Utilities

In This Chapter	8-3
Customizing ON-Bar and Storage-Manager Commands	8-3
Printing the Backup Boot Files	8-4
Migrating Backed-Up Logical Logs to Tape	8-6
Using start_worker.sh to Start onbar_worker Processes Manually.	8-7
Expiring and Synchronizing the Backup Catalogs	8-8
Choosing an Expiration Policy	8-9
Using the onmsync Utility	8-10
Removing Expired Backups	8-11
Expiring Old Backups on ISM	8-11
Regenerating the Emergency Boot File	8-12
Regenerating the sysutils Database	8-12
Deleting a Bad Backup	8-12
Expiring Backups Based on the Retention Date	8-12
Expiring a Generation of Backups	8-13
Expiring Backups Based on the Retention Interval	8-13
Expiring All Backups	8-13
Starting and Stopping ON-Bar Sessions.	8-14
Using the onbar_w Utility	8-15
Monitoring the Backup Scheduler Status	8-17
Using the onstat -g bus Option	8-17
Sample onstat -g bus Output Without Any ON-Bar Activity	8-17
Sample onstat -g bus Output During a Dbspace Backup	8-17
Using the onstat -g bus_sm Option	8-18
Sample onstat -g bus_sm Output When ON-Bar Is Idle.	8-18
Sample onstat -g bus_sm Output During a Dbspace Backup	8-19

In This Chapter

This chapter discusses the following topics:

- Customizing ON-Bar and storage-manager commands with the **onbar** script
- Starting **onbar-worker** processes manually
- Expiring and synchronizing the backup history
- Starting and stopping ON-Bar sessions
- Monitoring the backup scheduler status ♦

Customizing ON-Bar and Storage-Manager Commands

When you issue ON-Bar commands from the command line, the arguments are passed to the **onbar** script and then to **onbar_d**. Use the **onbar** shell script on UNIX or the **onbar** batch file on Windows to customize backup and restore commands, start ISM, and back up the ISM catalog. The **onbar** script is located in the **\$INFORMIXDIR/bin** directory on UNIX and in the **%INFORMIXDIR%\bin** directory on Windows.

The default **onbar** script assumes that the currently installed storage manager is ISM and backs up the ISM catalogs. If you are using a different storage manager, edit the **onbar** script, delete the ISM-specific lines, and optionally, add storage-manager commands.

For background information on the **onbar** script or batch file, see [“ON-Bar Utilities” on page 2-12](#) and [“Updating the onbar Script” on page 3-8](#).

IDS



UNIX

The default **onbar** script contains the following sections:

- Add start-up processing here

Use this section to initialize the storage manager, if necessary, and set environment variables.

- End start-up processing here

This section starts the **onbar_d** driver and checks the return code. Use this section for **onbar_d** and storage-manager commands.

- Add cleanup processing here

The code in this section backs up the ISM catalogs to the ISMData volume pool after the backup operation is complete. If you are using a third-party storage manager, delete the ISM-specific information.

If you use a name other than ISMData for the volume pool, change it to the name specified in the ISM_DATA_POOL configuration parameter.

The **archecker** temporary files are also removed. ♦

- End cleanup processing here

Use this section to return **onbar_d** error codes.

Warning: Edit the **onbar** script carefully. Accidental deletions or changes might cause undesired side effects. For example, backup verification might leave behind temporary files if the cleanup code near the end of the **onbar** script is changed.

Printing the Backup Boot Files

Use the following examples of what to add to the **onbar** script to print the emergency boot file if the backup is successful. Each time that you issue the **onbar -b** command, the emergency boot file is printed.

```
onbar_d "$@"      # receives onbar arguments from command line
return_code = $?  # check return code

# if backup (onbar -b) is successful, prints emergency boot file
if [$return_code -eq 0 -a "$1" = "-b"]; then
    servernum=`awk '/^SERVERNUM/ {print $2}' $INFORMIXDIR/etc/$ONCONFIG`
    lpr \${INFORMIXDIR}/etc/ixbar.$servernum
fi
exit $return_code
```

♦

XPS

To print the backup boot files on all coservers, replace the line:

```
lpr \${INFORMIXDIR}/etc/ixbar.$servernum
```

with:

```
xctl lpr \${INFORMIXDIR}/etc/Bixbar_\`hostname\`.$servernum
```

If more coservers than hosts are on the system, this script prints the same boot files twice. ♦

Windows

```
@echo off
%INFORMIXDIR%\bin\onbar_d %*
set onbar_d_return=%errorlevel%

if "%onbar_d_return%" == "0" goto backupcom
goto skip

REM Check if this is a backup command

:backupcom
if "%1" == "-b" goto printboot
goto skip

REM Print the onbar boot file

:printboot
print %INFORMIXDIR%\etc\ixbar.???

REM Set the return code from onbar_d (this must be on the last line of the
script)

:skip
%INFORMIXDIR%\bin\set_error %onbar_d_return%
:end
```

♦

UNIX

Migrating Backed-Up Logical Logs to Tape

You can set up your storage manager to back up logical logs to disk and then write a script to automatically migrate the logical logs from disk to tape for off-site storage. Edit the **onbar** script to call this migration script after the **onbar_d** process completes. The following example shows a script that calls the migration script:

```
onbar_d "$@" # starts the backup or restore
EXIT_CODE=$? # any errors?

PHYS_ONLY=false #if it's physical-only, do nothing
for OPTION in $*; do
    if [ $OPTION = -p ]; then
        PHYS_ONLY = true
    fi
done
if ! PHYS_ONLY; then # if logs were backed up, call another
    migrate_logs # program to move them to tape
fi
```

◆

Windows

This example for Windows invokes the migration script:

```
%INFORMIXDIR%\bin\onbar_d %*
set onbar_d_return=%errorlevel%

if "%onbar_d_return%" == "0" goto backupcom
goto skip

REM Check if the command is a backup command

:backupcom
if "%1" == "-b" goto m_log
if "%1" == "-l" goto m_log
goto skip

REM Invoke the user-defined program to migrate the logs

:m_log
migrate_log

REM Set the return code from onbar_d (this must be on the last line of the
script)

:skip
%INFORMIXDIR%\bin\set_error %onbar_d_return%

:end
```

◆

Using **start_worker.sh** to Start **onbar_worker** Processes Manually

To start **onbar-worker** processes manually, use the **start_worker.sh** script in `$INFORMIXDIR/etc` or the **onbar_w** utility. Edit this file to specify additional setup and cleanup tasks when the database server starts the **onbar-worker** processes or to set environment variables.

The default **start_worker.sh** script contains only one line, which calls **onbar_w** to start an **onbar-worker** process.

If the storage manager does not have special requirements for worker processes that pass data to it, you do not have to change the **start_worker.sh** script.

If the storage manager has special requirements, edit **start_worker.sh** to include operating-system commands that set up the environment before you start the **onbar-worker** process or perform other required actions after **onbar-worker** processes start.

The storage-manager documentation should describe any special requirements. If **onbar-worker** processes are not working correctly with a storage manager, check if the storage manager has any special requirements. If they are not listed in the documentation, or if they are not clearly stated so that you can add them to **start_worker**, contact the storage-manager manufacturer directly for more information.

Expiring and Synchronizing the Backup Catalogs

ON-Bar maintains a history of backup and restore operations in the **sysutils** database and an extra copy of the backup history in the emergency boot file. ON-Bar uses the **sysutils** database in a warm restore when only a portion of the data is lost. ON-Bar uses the emergency boot file in a cold restore because the **sysutils** database cannot be accessed.

To perform the following tasks, use the **onsmsync** utility:

- Expire old ON-Bar backups that you no longer need to restore to prevent unbounded growth in the emergency boot file and **sysutils** database
- Allow the storage manager to tell ON-Bar that a backup has expired according to its own internal policy. ON-Bar does not ask the storage manager to restore backups that are no longer available.
- Regenerate the emergency boot file

The **onsmsync** utility removes the following items from the **sysutils** database and the emergency boot file:

- Backups that the storage manager has expired
- Old backups based on the age of backup
- Old backups based on the number of times they have been backed up

Use **onsmsync** with the database server online or in quiescent mode to synchronize both the **sysutils** database and the emergency boot file. If the database server is offline, **onsmsync** synchronizes the storage manager with the emergency boot file.

To synchronize the sysutils database

1. Bring the database server online.
2. Run the **onmsync** utility without any options.

The **onmsync** utility synchronizes the **sysutils** database, the storage manager, and the emergency boot file as follows:

- Adds backup history to **sysutils** that is in the emergency boot file but is missing from the **sysutils** database.
- Removes the records of restores, whole-system restores, fake backups, successful and failed backups from the **sysutils** database.
- Expires old logical logs that are no longer needed.
- Regenerates the emergency boot file from the **sysutils** database.

Choosing an Expiration Policy

You can choose from the following three expiration policies:

- **Retention date (-t)** deletes all backups before a particular date and time.
- **Retention interval (-i)** deletes all backups older than some period of time.
- **Retention generation (-g)** keeps a certain number of versions of each backup.

ON-Bar always retains the latest level-0 backup for each storage space. It expires all level-0 backups older than the specified time unless they are required to restore from the oldest retained level-1 backup.

ON-Bar expires all level-1 backups older than the specified time unless they are required to restore from the oldest retained level-2 backup.

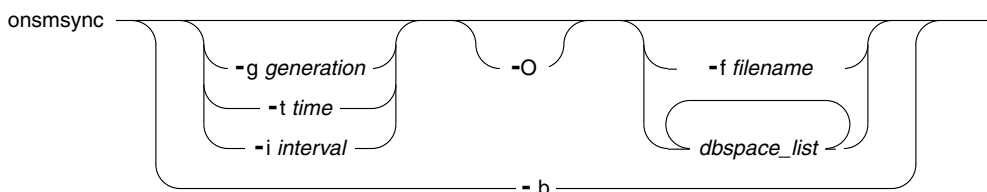
ON-Bar retains a whole-system backup that starts before the specified retention time and ends after the specified retention time. ♦



Using the onmsmsync Utility

The order of the commands does not matter except that the storage-space names or filename must come last.

Tip: To control whether the **sysutils** database maintains a history for expired backups and restores, use the **BAR_HISTORY** configuration parameter. For information, see [“BAR_HISTORY” on page 9-12](#).



Element	Purpose	Key Considerations
no options	Synchronizes the sysutils database and emergency boot file with the storage-manager catalog	None.
-b	Regenerates the emergency boot file from the sysutils database only	Not used with the other onmsmsync options. Does not synchronize with the storage manager.
dbspace_list	Lists the storage spaces to expire	If you enter more than one storage space, use a space to separate the names.
-f filename	Specifies the pathname of a file that contains a list of storage spaces to expire	Use this option to avoid entering a long list of storage spaces. The filename can be any valid UNIX or Windows filename.
-g generation	Retains a certain number of versions of each level-0 backup	The latest generation of backups are retained and all earlier ones are expired.

(1 of 2)

Element	Purpose	Key Considerations
-i interval	Expires all backups older than some period of time	Retains backups younger than this interval. Backups older than interval are not expired if they are needed to restore from other backups after that interval. Use the ANSI or GLS format for the <i>interval</i> : YYYY-MM or DD HH:MM:SS
-O	Enforces expiration policy strictly	If used with the -t , -g , or -i option, expires all levels of a backup, even if some of them are needed to restore from a backup that occurred after the expiration date. The -O option does not affect logical-log expiration. See “Expiring All Backups” on page 8-13 .
-t datetime	Expires all backups before a particular date and time	Retains backups younger than this datetime. Backups older than <i>datetime</i> are not expired if they are needed to restore from other backups after that <i>datetime</i> . Use the ANSI or GLS_DATETIME format for <i>datetime</i> .

(2 of 2)

Removing Expired Backups

If called with no options, the **onmsmsync** utility compares the backups in the storage-manager catalog with the backups in the **sysutils** database and emergency boot file. The **onmsmsync** utility expires and removes all extra backups from the **sysutils** database and emergency boot file.

The **onmsmsync** utility starts **onbar-merger** processes that delete backups on all nodes that contain storage managers. ♦

Expiring Old Backups on ISM

Important: ISM and certain third-party storage managers do not allow **onmsmsync** to delete backups from the storage manager. First, run **onmsmsync** without any parameters. Then, manually expire or delete the old backups from the storage manager.

1. To manually expire the old backups from ISM, use the **ism_config -volume name -retention #days** command.

For more information, see the *IBM Informix Storage Manager Administrator's Guide*.
2. Run **onmsmsync** without any options.

XPS



IDS

XPS



Regenerating the Emergency Boot File

To regenerate the emergency boot file only, use the following command:

```
onmsync -b
```

On Dynamic Server, this command saves the old emergency boot file as **ixbar.server_number.system_time** and regenerates it as **ixbar.server_number**. ♦

On Extended Parallel Server, this command saves the old backup boot file on each coserver as **Bixbar_hostname_system_time.server_number** and regenerates it as **Bixbar_hostname.server_number**. ♦

Regenerating the sysutils Database

If you lose the **sysutils** database, use the **bldutil** utility in **\$INFORMIXDIR/etc** on UNIX or **%INFORMIXDIR%\etc** on Windows to re-create the **sysutils** database with empty tables.

Then use the **onmsync** utility to re-create the backup and restore data in **sysutils**.

Important: If both the **sysutils** database and emergency boot file are missing, you cannot regenerate them with **onmsync**. Be sure to back up the emergency boot file with your other operating-system files.

Deleting a Bad Backup

The **onmsync** utility cannot tell which backups failed verification. If the *latest* backup failed verification but an earlier one was successful, you must manually delete the failed backup records from the storage manager and then run **onmsync** with no options to synchronize ON-Bar. For more information, see [Chapter 5, “Verifying Backups.”](#)

Expiring Backups Based on the Retention Date

The following example expires backups that started before November 24, 2000 and *all* fake backups, failed backups, and restores:

```
onmsync -t "2000-11-24 00:00"
```


Expiring a Generation of Backups

The following example retains the latest three sets of level-0 backups and the associated incremental backups, and expires *all* earlier backups and all restores, fake backups, and failed backups:

```
onmsync -g 3
```

Expiring Backups Based on the Retention Interval

The following example expires all backups that are older than three days and *all* fake backups, failed backups, and restores:

```
onmsync -i "3 00:00"
```

The following example expires all backups older than 18 months (written as 1 year + 6 months):

```
onmsync -i "1-6"
```

Expiring All Backups

The **onmsync** utility retains the latest level-0 backup unless you use the **-O** option. If you use the **-O** and **-t** options, all backups from before the specified *time* are removed even if they are needed for restore. If you use the **-O** and **-i** options, all backups from before the specified *interval* are removed even if they are needed for restore.

For example, to expire *all* backups, specify the following:

```
onmsync -O -g 0
```

Warning: If you use the **-O** option with the **-t**, **-i**, or **-g** options, you might accidentally delete critical backups, making restores impossible.



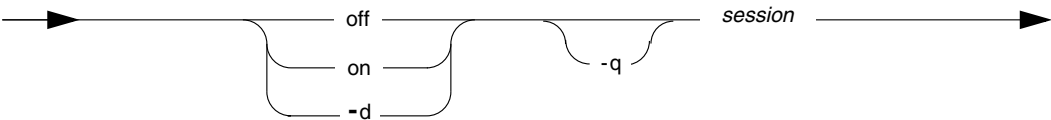
Starting and Stopping ON-Bar Sessions

Session control commands let you stop and restart ON-Bar sessions. You might stop and restart a session to:

- Temporarily stop continuous-log backup
- Temporarily stop some or all ON-Bar sessions while computer traffic is heavy

Starting and Stopping ON-Bar Sessions

Back to Syntax of ON-Bar Commands
p. 4-4



Element	Purpose	Key Considerations
off	Suspends a session	None.
on	Resumes a previously suspended session	None.
-d	Destroys a session	ON-Bar completes backing up the current storage spaces and logical logs. Storage spaces and logical logs that are not assigned to an onbar-worker are not processed.
-q session	Specifies the session name The -q option is optional but session is required.	The session name can be up to 128 characters. Put quotes around the session name if it contains white spaces.

The order of the options does not matter. For example, to destroy the ON-Bar session, **myses10**, use one of the following commands:

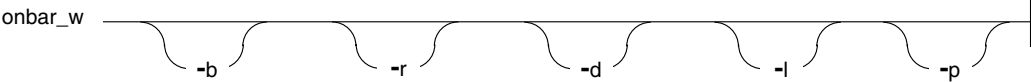
```
onbar -d -q myses10
onbar -q myses10 -d
```

Using the onbar_w Utility

To start **onbar-worker** processes manually, use the **onbar_w** utility.

[Using start_worker.sh to Start onbar_worker](#)

[Back to Syntax of ON-Bar Commands](#)
p. 4-4



Element	Purpose	Key Considerations
-b	Accepts request to back up storage spaces and logical logs	None.
-r	Accepts request to restore storage spaces and logical logs	None.
-d	Accepts request to back up and restore storage spaces	None.
-l	Accepts requests to back up and restore logical logs	None.
-p	Accepts queries from the Backup Scheduler that asks where particular objects have been backed up	None.

Specifying **onbar_w** without any options (the default) is the same as specifying **onbar_w -b -r -d -l -p**. [Figure 8-1](#) shows some examples of **onbar_w** options. Some option combinations also do the same thing.

Figure 8-1
onbar_w Options

onbar_w Options	What Happens
none	The onbar-worker process does everything. It backs up and restores storage spaces and logical logs, and places the storage spaces and logical logs in the correct storage manager.
-b	Backs up storage spaces and logical logs
-b -d	Backs up storage spaces
-b -l	Backs up logical logs
-b -p	Backs up storage spaces and logical logs, and places them in the storage manager
-b -r	Backs up and restores storage spaces and logical logs
-d -p	Backs up and restores storage spaces, and places them in the storage manager
-l	Backs up and restores logical logs
-p	Places storage spaces and logical logs in the storage manager
-r	Restores storage spaces and logical logs
-r -p	Restores storage spaces and logical logs, and places them in the storage manager
-b -r -d -l -p	Does everything
-b -r -p	Does everything
-d -l -p	Does everything

Monitoring the Backup Scheduler Status

Use the **onstat -g bus** and **onstat -g bus_sm** options to monitor the status of the Backup Scheduler. The Backup Scheduler tracks scheduled and active ON-Bar sessions. For SQL access to the Backup Scheduler, see [“Backup Scheduler SMI Tables” on page 10-11](#).

Using the onstat -g bus Option

The **onstat -g bus** option shows the current Backup Scheduler sessions, what work is scheduled for each ON-Bar session, and what work is currently in progress. Both options display identical information.

Sample onstat -g bus Output Without Any ON-Bar Activity

In the following example, two logical-log backup sessions are suspended:

```
onstat -g bus

Backup scheduler sessions
-----

Session "Log backup 2" state SUSPENDED error 0
Session "Log backup 1" state SUSPENDED error 0
```

Sample onstat -g bus Output During a Dbspace Backup

In the following example, ON-Bar and the Backup Scheduler are working on session **gilism824589**. Currently, dbspace **dbs1.2** is being backed up. Dbspaces **dbs12.1**, **dbs12.2**, and **other** are waiting to be backed up.

```
onstat -g bus

Backup scheduler sessions
-----

Session "Log backup 2" state SUSPENDED error 0
Session "Log backup 1" state SUSPENDED error 0
Session "gilism824589" state WAITING error 0
DBSPACE(dbs1.2) level 0 BACKUP,RUNNING
DBSPACE(dbs12.1) level 0 BACKUP,READY
DBSPACE(dbs12.2) level 0 BACKUP,READY
DBSPACE(other) level 0 BACKUP,READY
```

Using the *onstat -g bus_sm* Option

The *onstat -g bus_sm* option shows the current storage-manager configuration, what storage managers are assigned to each coserver, and what work each storage manager is currently performing.

Sample onstat -g bus_sm Output When ON-Bar Is Idle

The following example shows the storage-manager version, storage-manager name, the number of **onbar-worker** processes, the number of coservers, the maximum number of **onbar-worker** processes started, and the ON-Bar idle time-out:

```
onstat -g bus_sm

Configured storage managers
-----

BAR_SM                1
BAR_SM_NAME           ism
BAR_WORKER_COSVR      1
BAR_DBS_COSVR         1,2
BAR_LOG_COSVR         1,2
BAR_WORKER_MAX        1
BAR_IDLE_TIMEOUT      5
END
```

Sample *onstat -g bus_sm* Output During a Dbospace Backup

When a backup or restore session is active, **onstat -g bus_sm** also displays information about the active **onbar-worker** processes, as the following example shows:

```
onstat -g bus_sm

Configured storage managers
-----

BAR_SM                1
BAR_SM_NAME           ism
BAR_WORKER_COSVR      1
BAR_DBS_COSVR         1,2
BAR_LOG_COSVR         1,2
BAR_WORKER_MAX        1
BAR_IDLE_TIMEOUT      5
END

Active workers:

Worker 2 Coserver 1 Pid 4590 State BUSY "dbsl.2.0"
```

The **onbar-worker** process is backing up dbslice **dbsl.2.0**.

Setting ON-Bar Configuration Parameters

In This Chapter	9-3
Setting archchecker Configuration Parameters in AC_CONFIG	9-4
AC_CONFIG File	9-4
AC_MSGPATH	9-5
AC_STORAGE	9-5
AC_VERBOSE	9-6
Setting ON-Bar Parameters in ONCONFIG	9-7
ALARMPROGRAM	9-7
BAR_ACT_LOG	9-8
BAR_BOOT_DIR	9-9
BAR_BSAIB_PATH	9-10
BAR_DBS_COSVR	9-11
BAR_HISTORY	9-12
BAR_IDLE_TIMEOUT	9-13
BAR_LOG_COSVR	9-13
BAR_MAX_BACKUP	9-14
BAR_NB_XPORT_COUNT.	9-15
BAR_PROGRESS_FREQ	9-16
BAR_RETRY.	9-17
BAR_SM	9-19
BAR_SM_NAME	9-19
BAR_WORKER_COSVR	9-20
BAR_WORKER_MAX	9-20
BAR_XFER_BUF_SIZE	9-22
BAR_XFER_BUFSIZE	9-23
BAR_XPORT_COUNT	9-24
ISM_DATA_POOL	9-24

ISM_LOG_POOL	9-25
LOG_BACKUP_MODE	9-25
LTAPEDEV	9-26
RESTARTABLE_RESTORE	9-27
Files That ON-Bar and ISM Use	9-28

In This Chapter

This chapter describes the ON-Bar configuration parameters that you can set in the ONCONFIG file and the **archecker** configuration parameters that you can set in the AC_CONFIG file.

Be sure to configure your storage manager. Depending on the storage manager that you choose, you might set different ON-Bar configuration parameters. Before you start ON-Bar, see [“Installing and Configuring a Third-Party Storage Manager” on page 3-3](#).

This table describes the following attributes (if relevant) for each parameter.

Attribute	Description
<i>onconfig.std value</i>	The default value that appears in the onconfig.std file in Dynamic Server or the onconfig.xps file in Extended Parallel Server
<i>if not present</i>	<p>The value that the database server supplies if the parameter is missing from your ONCONFIG file</p> <p>If this value is present in onconfig.std, the database server uses the onconfig.std value. If this value is not present in onconfig.std, the database server uses this value.</p>
<i>units</i>	The units in which the parameter is expressed
<i>range of values</i>	The valid values for this parameter
<i>takes effect</i>	<p>The time at which a change to the value of the parameter affects ON-Bar operation</p> <p>Except where indicated, you can change the parameter value between a backup and a restore.</p>
<i>refer to</i>	Cross-reference to further discussion

Setting archecker Configuration Parameters in AC_CONFIG

Because ON-Bar calls the **archecker** utility to verify backups, you must configure these **archecker** parameters before you can use the **onbar -v** option. For more information about using **archecker**, see [Chapter 5, “Verifying Backups.”](#) The following table shows the **archecker** configuration parameters that you specify in the AC_CONFIG file.

Configuration Parameter	Purpose	IDS	XPS
AC_MSGPATH	Specifies the location of the archecker message log	✓	✓
AC_STORAGE	Specifies the location of the temporary files that archecker builds	✓	✓
AC_VERBOSE28	Specifies either verbose or quiet mode for archecker messages	✓	✓

AC_CONFIG File

onconfig.std value UNIX \$INFORMIXDIR/etc/ac_config.std
 Windows %INFORMIXDIR%\etc\ac_config.std

takes effect When **onbar** starts

Set the AC_CONFIG environment variable to the full pathname for the **archecker** configuration file (either **ac_config.std** or user defined). You must specify the entire path, including the configuration filename in the AC_CONFIG file or else the **archecker** utility might not work correctly. The following table shows examples of valid AC_CONFIG pathnames.

/usr/informix/etc/ac_config.std (UNIX)	/usr/local/my_ac_config.std
c:\Informix\etc\ac_config.std (Windows)	c:\Informix\etc\my_ac_config.std

If **AC_CONFIG** is not set, the **archecker** utility sets the default location for the archecker configuration file to **\$INFORMIXDIR/etc/ac_config.std** on UNIX or **%INFORMIXDIR%\etc\ac_config.std** on Windows.

AC_MSGPATH

onconfig.std value	UNIX	/tmp/ac_msg.log
	Windows	c:\temp\ac_msg.log
takes effect	When onbar starts	

The **AC_MSGPATH** parameter in the **AC_CONFIG** file specifies the location of the **archecker** message log (**ac_msg.log**). You must specify the entire path of the message log in the **AC_CONFIG** file or else the **archecker** utility might not work correctly.

When you verify backups with **onbar -v**, the **archecker** utility writes summary messages to the **bar_act.log** and indicates whether the validation succeeded or failed. It writes detailed messages to the **ac_msg.log**. If the backup fails verification, discard the backup and retry another backup, or give the **ac_msg.log** to Technical Support. For sample messages, see [“Interpreting Verification Messages” on page 5-11](#).

AC_STORAGE

onconfig.std value	UNIX	/tmp
	Windows	c:\temp
takes effect	When onbar starts	

The **AC_STORAGE** parameter in the **AC_CONFIG** file specifies the location of the directory where **archecker** stores its temporary files. You must specify the entire path of the storage location in **AC_CONFIG** or else the **archecker** utility might not work correctly.

Figure 9-1 lists the directories and files that **archecker** builds. If verification is successful, these files are deleted.

Figure 9-1
archecker Temporary Files

Directory	Files
CHUNK_BM	Bitmap information for every backed up storage space.
INFO	Statistical analysis and debugging information for the backup.
SAVE	Partition pages in the PT.##### file. Chunk-free pages in the FL.##### file. Reserved pages in the RS.##### file. Blob-free map pages in the BF.##### file (IDS only).

To calculate the amount of free space that you need, see [“Estimating the Amount of Temporary Space for archecker” on page 5-8](#). We recommend that you set AC_STORAGE to a location with plenty of free space.

AC_VERBOSE

onconfig.std value	1
range of values	1 for verbose messages in ac_msg.log 0 for terse messages in ac_msg.log
takes effect	When onbar starts

The AC_VERBOSE parameter in the AC_CONFIG file specifies either verbose or terse output in the **archecker** message log.



Setting ON-Bar Parameters in ONCONFIG

The following section lists the ON-Bar configuration parameters and indicates the database servers to which they apply.

Important: ON-Bar does not use the `TAPEDEV`, `TAPEBLK`, `TAPESIZE`, `LTAPEBLK`, and `LTAPESIZE` configuration parameters. ON-Bar checks if `LTAPEDEV` is set to `/dev/null` on UNIX or `\dev\nul` or `NUL` on Windows. For more information, see “[LTAPEDEV](#)” on page 9-26.

ALARMPROGRAM

<i>onconfig.std value</i>	Dynamic Server: On UNIX: <code>/usr/informix/etc/log_full.sh</code> On Windows: <code>%INFORMIXDIR%\etc\log_full.bat</code> Extended Parallel Server: blank
<i>if value not present</i>	Dynamic Server: <code>no_log.sh</code> or <code>no_log.bat</code> Extended Parallel Server: blank
<i>range of values</i>	Full pathname
<i>takes effect</i>	When the database server is shut down and restarted
<i>refer to</i>	<i>Administrator's Reference</i>

Use the ALARMPROGRAM configuration parameter to handle event alarms and start or end automatic log backups.

Use the shell script, `log_full.sh` or `log_full.bat`, for starting automatic log backups. Modify this script and set it to the full path of ALARMPROGRAM in the ONCONFIG file.

To generate event alarms, set ALARMPROGRAM to `$INFORMIXDIR/etc/alarmprogram.sh` or `%INFORMIXDIR%\etc\alarmprogram.bat` and modify the file according. For more information, see the *Administrator's Reference*.



Important: When you choose automatic logical-log backups, backup media should always be available for the backup process.

Do not use the continuous log backup command (**onbar -b -l -C**) if you have automatic log backup set up through the **ALARMPROGRAM** parameter, and conversely.

BAR_ACT_LOG

takes effect

IDS: When **onbar-driver** starts

XPS: When **onbar** starts or an **onbar-worker** process starts

The **BAR_ACT_LOG** configuration parameter specifies the full pathname of the ON-Bar activity log. You should specify a path to an existing directory with an appropriate amount of space available or use **\$INFORMIXDIR/bar_act.log**

Whenever a backup or restore activity or error occurs, ON-Bar writes a brief description to the activity log. The format of the file resembles the format of the database server message log. You can examine the activity log to determine the results of ON-Bar actions.

Specifying BAR_ACT_LOG with a Filename Only

If you specify a filename only in the **BAR_ACT_LOG** parameter, ON-Bar creates the ON-Bar activity log in the working directory in which you started ON-Bar. For example, if you started ON-Bar from **/usr/mydata** on UNIX, the activity log is written to that directory.

UNIX

If the database server launches a continuous logical-log backup, ON-Bar writes to the ON-Bar activity log in the working directory for the database server. ♦

Windows

If the database server launches a continuous logical-log backup, ON-Bar writes to the activity log in the **%INFORMIXDIR%\bin** directory instead. ♦

XPS



XPS

Using BAR_ACT_LOG on Extended Parallel Server

The **onbar_w** and **onbar_m** utilities also write messages to the activity log.

If you set **BAR_ACT_LOG** to a local pathname, each **onbar-driver** and **onbar-worker** process writes to the ON-Bar activity log on the node where it runs. Unless your system is configured to have shared directories, and the ON-Bar activity log is specified to be in a shared directory, each node has its own ON-Bar activity log.

Warning: Even if you set the path of **BAR_ACT_LOG** to some other directory, check to see if the ON-Bar activity log was placed in the default directory. When **onbar-merger** first starts, it writes messages to **/tmp/bar_act.log** until it has a chance to read the **ONCONFIG** file.

BAR_BOOT_DIR

onconfig.std value **/usr/informix/etc**

if value not present blank

takes effect When the next **onbar-worker** process starts

The **BAR_BOOT_DIR** configuration parameter allows you to change the directory path for the emergency boot files. You must specify it in the global section of the **ONCONFIG** file.

Accessing the Emergency Boot Files

The **onbar_w**, **onbar_m**, and **onbar_d** utilities must be able to access the directory that you select for the emergency boot files. All coservers must be able to write to the directory for user root. The **BAR_BOOT_DIR** parameter is designed to be used on a system where the **INFORMIXDIR** directory is shared but the root user does not have write permissions to the **INFORMIXDIR** directory. For example, you can set **BAR_BOOT_DIR** to the **/var/informix** directory on a multinode system. Because each node has its own **/var** file system, it prevents read/write permissions problems.

If you change the **BAR_BOOT_DIR** value while an **onbar-worker** process is running, it will use the old value. Any new **onbar-worker** process started after the change will pick up the new value.



Warning: After you change the `BAR_BOOT_DIR` value, you must kill all *onbar-worker* processes on all coservers. When the processes end, either perform a level-0 backup of all storage spaces or move the files from the *etc* subdirectory to the new location.

Saving the Emergency Boot Files During Reversion

When you revert from Version 8.40 to an earlier database server version, the reversion script saves the emergency boot files in the directory that the `BAR_BOOT_DIR` parameter specifies. Otherwise, the reversion script saves the emergency boot files to the `$INFORMIXDIR/etc` directory if `BAR_BOOT_DIR` is not set in the `ONCONFIG` file or is set to an invalid directory.

BAR_BSALIB_PATH

onconfig.std value	UNIX	<code>\$INFORMIXDIR/lib/ibsad001.platform_</code> <i>extension</i> where <i>platform_extension</i> is the shared- library file extension
	Windows	<code>libbsa.dll</code>
if value not present	Windows	None
takes effect	IDS:	When onbar-driver starts
	XPS:	When an onbar-worker process starts

ON-Bar and the storage manager rely on a shared library to integrate with each other. Configure the `BAR_BSALIB_PATH` configuration parameter for your storage-manager library. Support for `BAR_BSALIB_PATH` is platform-specific. Check your machine notes to determine if you can use it with your operating system. You can change the value of `BAR_BSALIB_PATH` between a backup and restore.

UNIX

To ensure that this integration takes place, specify the shared-library pathname. Set one of the following options:

- Place the storage-manager library in the default directory.
For example, the suffix for Solaris is **so**, so you specify **\$INFORMIXDIR/lib/ibsad001.so** on a Solaris system.
- Place the storage-manager library in any directory and create a symbolic link from **\$INFORMIXDIR/lib/ibsad001.platform_extension** to it.
If you use ISM, create a symbolic link to **\$INFORMIXDIR/lib/libbsa.platform_extension** or set **BAR_BSALIB_PATH** to this absolute path value.
- Set the **LD_LIBRARY_PATH** environment variable. For example, to use ISM on Solaris, set **LD_LIBRARY_PATH** to **\$INFORMIXDIR/lib**. ♦
- Set the pathname for the ISM shared library to **%ISMDIR%\bin\libbsa.dll**.

Windows

The **%ISMDIR%** variable includes a version or release number. For example: **set ISMDIR=C:\program files\informix\ism\2.20**. This directory is set when the database server is installed on Windows. This pathname is different if you use a different storage manager. ♦

Tip: Be sure that the shared library can access the backup data in the storage manager in a restore. You cannot back up using one storage manager and restore using a different storage manager.



XPS

BAR_DBS_COSVR

onconfig.std value all coservers

range of values A list of unique positive integers greater than or equal to one

takes effect When the database server starts

The optional **BAR_DBS_COSVR** configuration parameter specifies the coservers from which the storage manager that **BAR_SM** specifies can be sent storage-space backup and restore data. If **BAR_DBS_COSVR** is set to 0, the storage manager is not given dbspaces from any coserver. You might specify **BAR_DBS_COSVR 0** to reserve a storage manager for logical-log backups only.

The values are coserver numbers, separated by commas. Hyphens indicate ranges. For example, BAR_DBS_COSVR 1-5,7,9 specifies a storage manager that backs up dbspaces on coservers 1, 2, 3, 4, 5, 7, and 9. Do not put spaces between coserver names.

To provide flexibility and improved performance, this list of coservers can overlap with values listed for other storage managers.

BAR_HISTORY

onconfig.std value	Not in onconfig.std
<i>if value not present</i>	0
<i>range of values</i>	0 to remove records for expired backup objects from sysutils 1 to keep records for expired backup objects in sysutils
<i>takes effect</i>	When onsmsync starts

The BAR_HISTORY parameter specifies whether the **sysutils** database maintains a backup history when you use **onsmsync** to expire old backups. For more information, see [“Using the onsmsync Utility” on page 8-10](#).

If you set the value to 0, **onsmsync** removes the **bar_object**, **bar_action**, and **bar_instance** rows for the expired backup objects from the **sysutils** database. If you set the value to 1, **onsmsync** sets the **act_type** value to 7 in the **bar_action** row and keeps the **bar_action** and **bar_instance** rows for expired backup objects in the **sysutils** database. If you do not set BAR_HISTORY to 1, the restore history is removed.

Regardless of the value of BAR_HISTORY, **onsmsync** removes the line that describes the backup object from the emergency boot file and removes the object from the storage manager when the storage manager expires the object.

XPS

BAR_IDLE_TIMEOUT**onconfig.std value** 0*if value not present* 0**units** Minutes**range of values** 0 to unlimited**takes effect** When the database server starts

The BAR_IDLE_TIMEOUT configuration parameter determines the maximum amount of time in minutes that an **onbar-worker** process can be idle before it is shut down.

The BAR_IDLE_TIMEOUT configuration parameter is optional. If BAR_IDLE_TIMEOUT is set to 0, the **onbar-worker** processes never shut down until you shut down the database server.

You can set this option locally for individual storage managers to override the default or specified global setting.

XPS

BAR_LOG_COSVR**onconfig.std value** all coservers**range of values** A list of unique positive integers greater than or equal to one**takes effect** When the database server starts

The BAR_LOG_COSVR configuration parameter specifies the coservers from which the storage manager that BAR_SM specifies can be sent logical-log backup and restore data. BAR_LOG_COSVR is optional. The default is all coservers.

If BAR_LOG_COSVR is set to 0, the storage manager is not given logical logs from any coserver. You might specify BAR_LOG_COSVR 0 to reserve a storage manager for dbspace backups only.

The values are coserver numbers, separated by commas. Hyphens indicate ranges. For example, BAR_LOG_COSVR 1-5,7,9 specifies a storage manager that backs up logical logs on coservers 1, 2, 3, 4, 5, 7, and 9. Do not put spaces between coserver names.

Extended Parallel Server restricts BAR_LOG_COSVR settings to guarantee that no two storage-manager instances can back up logs for the same coserver.

BAR_MAX_BACKUP

onconfig.std value 0

if value not present 4

units **onbar** processes

range of values 0 = maximum number of processes allowed on system
1 = serial backup or restore
n = specified number of processes spawned

takes effect When **onbar** starts

The BAR_MAX_BACKUP parameter specifies the maximum number of parallel processes that are allowed for each **onbar** command. Both UNIX and Windows support parallel backups. Although the database server default value for BAR_MAX_BACKUP is 4, the **onconfig.std** value is 0.

Specifying Serial Backups and Restores

To perform a serial backup or restore, set BAR_MAX_BACKUP to 1.

ON-Bar ignores the BAR_MAX_BACKUP parameter for a whole-system backup because they are always done serially.

Specifying Parallel Backups and Restores

To specify parallel backups and restores, set BAR_MAX_BACKUP to a value other than 1. For example, if you set BAR_MAX_BACKUP to 5 and execute an ON-Bar command, the maximum number of processes that ON-Bar will spawn concurrently is 5. Configure BAR_MAX_BACKUP to any number up to the maximum number of storage devices or the maximum number of streams available for physical backups and restores.

If you set BAR_MAX_BACKUP to 0, the system creates as many ON-Bar processes as needed. The number of ON-Bar processes is limited only by the number of storage spaces or the amount of memory available to the database server, whichever is less.

The amount of memory available is based on SHMTOTAL. ON-Bar performs the following calculation where N is the maximum number of ON-Bar processes that are allowed:

$$N = \text{SHMTOTAL} / (\# \text{ transport buffers} * \text{size of transport buffers} / 1024)$$

If SHMTOTAL is 0, BAR_MAX_BACKUP is reset to 1. If N is greater than BAR_MAX_BACKUP, ON-Bar uses the BAR_MAX_BACKUP value. Otherwise, ON-Bar starts N backup or restore processes.

IDS

BAR_NB_XPORT_COUNT

<i>onconfig.std value</i>	10
<i>if value not present</i>	10
<i>units</i>	Buffers
<i>range of values</i>	3 to unlimited
<i>takes effect</i>	When onbar starts

The BAR_NB_XPORT_COUNT configuration parameter specifies the number of data buffers that each **onbar_d** process can use to exchange data with the database server. The value of this parameter affects **onbar** performance. For example, if you set BAR_MAX_BACKUP to 5 and BAR_NB_XPORT_COUNT to 5 and subsequently issue 5 ON-Bar commands, the resulting 25 ON-Bar processes will use a total of 125 buffers.

To calculate the amount of memory that each **onbar_d** process requires, use the following formula. For information on the page size for your system, see the release notes.

$$\text{required_memory} = (\text{BAR_NB_XPORT_COUNT} * \text{BAR_XFER_BUF_SIZE} * \text{page_size}) + 5 \text{ MB}$$

BAR_PROGRESS_FREQ

onconfig.std value 0

if value not present 0

units Minutes

range of values 0, then 5 to unlimited

takes effect When **onbar** starts

The BAR_PROGRESS_FREQ configuration parameter specifies, in minutes, the frequency of the progress messages in the ON-Bar activity log for backup and restore operations. For example, if you set BAR_PROGRESS_FREQ to 5, ON-Bar reports the percentage of the object backed up or restored every five minutes. If you set BAR_PROGRESS_FREQ to 0 or do not set it, ON-Bar does not write any progress messages to the activity log.

Too frequent progress messages fill the activity log. So if you set BAR_PROGRESS_FREQ to 1, 2, 3, or 4, ON-Bar automatically resets it to 5 to prevent overflow in the ON-Bar activity log. Infrequent progress messages might make it hard to tell if the operation is progressing satisfactorily.

If ON-Bar cannot determine the size of the backup or restore object, it reports the number of transfer buffers sent to the database server instead of the percentage of the object backed up or restored.

BAR_RETRY

The BAR_RETRY configuration parameter specifies how many times **onbar** should retry a data backup, logical-log backup, or restore operation if the first attempt fails.

BAR_RETRY on Dynamic Server

<i>onconfig.std value</i>	1
<i>if value not present</i>	1
<i>range of values</i>	BAR_ABORT (0), BAR_CONT (1), or n
<i>takes effect</i>	When onbar starts

The setting of the BAR_RETRY parameter determines ON-Bar behavior in the following ways:

- If set to BAR_ABORT, ON-Bar aborts the backup or restore session when an error occurs for a storage space or logical log, returns an error, and quits. If ON-Bar is running in parallel, the already running processes finish but no new ones are started.
- If set to BAR_CONT, ON-Bar aborts the backup or restore attempt for that particular storage space, returns an error, and attempts to back up or restore any storage spaces or logical logs that remain.
- If set to a specific number (n), ON-Bar attempts to back up or restore this storage space or logical log the specified number of times before it gives up and moves on to the next one.

BAR_RETRY on Extended Parallel Server

<i>onconfig.std value</i>	0. Does not retry
<i>range of values</i>	0 to 5
<i>takes effect</i>	When the database server starts

If a backup or restore fails, ON-Bar attempts to back up or restore the object the specified number of times before it gives up and moves on to the next object.

The Backup Scheduler maintains two retry counts: object retries and storage-manager retries.

Object retries is the number of times that the Backup Scheduler attempts a backup or restore operation. If the backup or restore of a particular object gets an error, the Backup Scheduler retries it BAR_RETRY times. If it continues to fail, the Backup Scheduler removes the object from the backup session.

To restart the backup or restore operation

1. Resolve the error.
2. Issue another ON-Bar command to back up or restore that object.

Storage-manager retries is the number of times that the Backup Scheduler attempts to start an **onbar-worker** process before giving up. If an **onbar-worker** process registers with the Backup Scheduler but exits with an error, then the Backup Scheduler tries to start another **onbar-worker** process, up to BAR_RETRY times. If the **onbar-worker** process fails before it registers or if the Backup Scheduler already has tried to start an **onbar-worker** process BAR_RETRY times, it does not start another **onbar-worker** process.

To reset the storage-manager retry counter and restart the operation

1. Resolve the error.
2. Start an **onbar-worker** process manually, either directly on the command line or by calling **start_worker.sh** on UNIX or **start_worker.bat** on Windows. The backup or restore operation can then resume.

You can monitor the storage-manager retry count with **onstat -g bus_sm**. For more information, see [“Monitoring the Backup Scheduler Status” on page 8-17](#).

XPS

BAR_SM

onconfig.std value 1

range of values Positive integer greater than or equal to 1

takes effect When the database server starts

The required BAR_SM configuration parameter identifies a specific storage-manager instance. Only ON-Bar and the Backup Scheduler use this value. The storage manager does not use this value.

The system uses the BAR_SM to track the location of backups. If you change the identification number after you use the storage manager to perform a backup, you invalidate the backups that you have made. Perform a new level-0 backup of all data.

XPS

BAR_SM_NAME

onconfig.std value None

range of values Any character string that does not contain white spaces or the pound sign (#)

takes effect When the database server starts

The BAR_SM_NAME configuration parameter is the name of the storage manager. It can be up to 128 characters.

XPS

BAR_WORKER_COSVR**onconfig.std value** 1**if value not present** an error occurs**range of values** A list or range of unique positive integers greater than or equal to one**takes effect** When the database server starts

The required BAR_WORKER_COSVR configuration parameter specifies which coservers can access the storage manager that BAR_SM identifies. If BAR_SM is specified, BAR_WORKER_COSVR must also be specified. Any coserver on the list can restore data that other coservers on the list back up.

Enter the numbers of the coservers where **onbar-worker** processes can be started for this storage manager. If you enter only one coserver number, use the number of a coserver on a node with a physically attached storage device so that the storage manager does not have to transfer data across the network. This step improves performance.

The list must not overlap with the list of any other storage manager. The values are coserver numbers, separated by commas. Hyphens indicate ranges. For example, BAR_WORKER_COSVR 1-3,5,7 specifies a storage manager that can access **onbar-worker** processes running on coservers 1, 2, 3, 5, and 7. Do not put spaces between coserver names.

XPS

BAR_WORKER_MAX**onconfig.std value** 0**if value not present** 0**takes effect** When the database server starts

The BAR_WORKER_MAX configuration parameter determines how many storage spaces and logical logs can be backed up or restored in parallel on each storage manager. It also specifies the maximum number of **onbar-worker** processes that the database server automatically starts for this storage-manager instance. If the BAR_WORKER_MAX value is 0, you must start the **onbar-worker** processes manually.

Specifying a Parallel Backup or Restore

For example, to start five **onbar-worker** processes in parallel, set BAR_WORKER_MAX to 5. Set BAR_WORKER_MAX to the number of storage devices available to the storage manager. If the database server has multiple storage managers configured, the number of parallel operations is the sum of BAR_WORKER_MAX values for all storage managers. The maximum number of **onbar-worker** processes that run simultaneously depends on the capabilities of the storage manager.

If **onbar-worker** processes for a specific storage manager have special start-up requirements, such as environment variables, you can specify these by editing the **start_worker** script file. For information, see [“Using start_worker.sh to Start onbar_worker Processes Manually”](#) on page 8-7. If storage managers have dynamic requirements for **onbar-worker** processes, you might have to start them manually.

Specifying a Serial Backup or Restore

You can specify serial backups and restores with Extended Parallel Server in two ways:

- Set BAR_WORKER_MAX to 0 and manually start one **onbar-worker** process.
You must start the **onbar-worker** processes manually before you can back up or restore data.
- Set BAR_WORKER_MAX to 1.
The database server starts one **onbar-worker** process and backs up or restores the data serially for this storage-manager instance.

BAR_XFER_BUF_SIZE

onconfig.std value 31

units page size

range of values 1 to 15 pages when the PAGESIZE is 4 kilobytes
1 to 31 when the PAGESIZE is 2 kilobytes

The maximum buffer size is 64 kilobytes, so
BAR_XFER_BUF_SIZE * pagesize <= 64 kilobytes

takes effect When onbar starts

The BAR_XFER_BUF_SIZE configuration parameter specifies the size of each transfer buffer. The database server passes the buffer to ON-Bar and the storage manager. To calculate the size of the transfer buffer in a storage space or logical-log backup, use the following formula:

$$\text{transfer buffers} = \text{BAR_XFER_BUF_SIZE} * \text{pagesize}$$

To calculate how much memory the database server needs, use the following formula:

$$\text{memory} = (\text{BAR_XFER_BUF_SIZE} * \text{PAGESIZE}) + 500$$

The extra 500 bytes is for overhead. For example, if BAR_XFER_BUF_SIZE is 15, the transfer buffer should be 61,940 bytes.

Important: You cannot change the buffer size between a backup and restore.



XPS

BAR_XFER_BUFSIZE

onconfig.std value 8 pages

if value not present 32,768 / pagesize for platform

units PAGESIZE which can be 2, 4, or 8 kilobytes

range of values 1 to 15 pages

takes effect When the **onbar_w** utility starts

The BAR_XFER_BUFSIZE configuration parameter specifies the size of each transfer buffer. The actual size of a transfer buffer is $\text{BAR_XFER_BUFSIZE} * \text{PAGESIZE} + 500$. The extra 500 is for overhead. For example, if BAR_XFER_BUFSIZE is 15 and the page size is 4 kilobytes, the transfer buffer should be 61,440 bytes.

For generally good performance, set to 8, although different storage managers might suggest other values. The maximum value that XBSA allows is 64 kilobytes.

If you set BAR_XFER_BUFSIZE to 8 when the page size is 8 kilobytes, the database server decrements the transfer buffer size to 7 pages so that the maximum value would be under 64 kilobytes ($7 * 8 = 56$). ON-Bar displays a warning message if the transfer buffer size is decremented.

To calculate how much memory the database server needs, use the following formula:

$$\text{kilobytes} = \text{BAR_WORKER_MAX} * \text{BAR_XFER_BUFSIZE} * \text{BAR_XPORT_COUNT} * \text{PAGESIZE}$$

For example, you would need at least 1600 kilobytes of memory for the transfer buffers for five **onbar-worker** processes if the page size is 4 kilobytes.

$$1600 \text{ kilobytes} = 5 \text{ workers} * 8 \text{ pages} * 10 \text{ buffers} * 4 \text{ kilobytes}$$

Important: You cannot change the buffer size (value of BAR_XFER_BUFSIZE) between a backup and restore.



XPS

BAR_XPORT_COUNT

onconfig.std value	10
if value not present	10
units	Buffers
range of values	3 to unlimited
takes effect	When an onbar-worker process starts

The BAR_XPORT_COUNT configuration parameter specifies the number of data buffers that each **onbar-worker** process can use to exchange data with Extended Parallel Server. The value of this parameter affects **onbar-worker** performance.

To calculate the amount of memory that each **onbar_w** process requires, use the following formula:

$$\text{required_memory} = (\text{BAR_XPORT_COUNT} * \text{BAR_XFER_BUFSIZE} * \text{page_size}) + 5 \text{ MB}$$

For information on configuring the page size, see the *Administrator's Guide*.

ISM_DATA_POOL

onconfig.std value	ISMDData
takes effect	When onbar starts

The ISM_DATA_POOL parameter, when listed in the **ONCONFIG** file for the database server, specifies the volume pool that you use for backing up storage spaces. The value for this parameter can be any volume pool that ISM recognizes. If this parameter is not present, ISM uses the ISMDData volume pool. For details, see the *IBM Informix Storage Manager Administrator's Guide*.

XPS

Insert the ISM_DATA_POOL parameter inside the BAR_SM paragraph in the storage-manager section of the **ONCONFIG** file if you want it to apply to one storage-manager instance. Insert ISM_DATA_POOL in the global section of the **ONCONFIG** file if you want it to apply to all storage-manager instances. ♦

For more information, see [“Files That ON-Bar and ISM Use” on page 9-28](#).

ISM_LOG_POOL

onconfig.std value ISMLogs
takes effect When **onbar** starts

The ISM_LOG_POOL parameter, when listed in the **ONCONFIG** file for the database server, specifies the volume pool that you use for backing up logical logs. The value for this parameter can be any volume pool that ISM recognizes. If this parameter is not present, ISM uses the ISMLogs volume pool. For details, see the *IBM Informix Storage Manager Administrator's Guide*.

Insert the ISM_LOG_POOL parameter inside the BAR_SM paragraph in the storage-manager section of the **ONCONFIG** file if you want it to apply to one storage-manager instance. Insert ISM_LOG_POOL in the global section of the **ONCONFIG** file if you want it to apply to all storage-manager instances. ♦

For more information, see [“Files That ON-Bar and ISM Use” on page 9-28](#).

XPS

LOG_BACKUP_MODE

Use the LOG_BACKUP_MODE configuration parameter to determine how logical-log files are backed up after they fill.

onconfig.std value MANUAL

<i>range of values</i>	CONT	Continuous = an onbar-worker process backs up each logical-log file as soon as it fills.
	MANUAL	Manual = queues the logical-log files until you can issue an onbar -b -l command.
	NONE	Use the NONE option if you do not want to back up the logs. The database server marks the logical logs as backed up as soon as they are full so that ON-Bar cannot back them up. When the database server starts up, it writes a message to the online.log if LOG_BACKUP_MODE = NONE.

takes effect When the database server restarts

XPS



IDS



Warning: If you set `LOG_BACKUP_MODE` to `NONE`, you cannot back up logs. All transactions in those logs are lost, and you will not be able to restore them. Your physical backups also will not be restorable.

LTAPEDEV

If you specify a tape device in the `LTAPEDEV` configuration parameter, ON-Bar ignores the value. ON-Bar also ignores the value of the `LTAPEBLK`, `LTAPESIZE`, `TAPEDEV`, `TAPEBLK`, and `TAPESIZE` parameters. Consider leaving these parameter values blank when you use ON-Bar because the storage manager sets these values.

Warning: Set `LTAPEDEV` to `/dev/null` or leave it blank on UNIX or `\dev\nul` or `NUL` on Windows only if you do **not** want to back up the logical logs. The ON-Bar activity log will display a warning and return code 152. Because the database server marks the logical logs as backed up when they are no longer current, ON-Bar cannot find logical logs to back up. All transactions in those logs are lost, and you will not be able to restore them.

If you performed a whole-system backup with `LTAPEDEV` set to null, you must use the **`onbar -r -w -p`** command during restore to notify ON-Bar that you do not want to restore the logs. For more information, see [“Considerations When LTAPEDEV is Set to Null” on page 6-21](#).

RESTARTABLE_RESTORE

Use the RESTARTABLE_RESTORE configuration parameter to enable or disable restartable restores.

onconfig.std value ON

if value not present ON

<i>range of values</i>	OFF	Disables restartable restore. If a restore fails and RESTARTABLE_RESTORE is OFF, you will not be able to restart it.
	ON	Enables restartable restore. Set RESTARTABLE_RESTORE to ON before you begin a restore. Otherwise, you will be unable to restart the restore after a failure.

takes effect If you need to restart a physical restore, you do not need to restart the database server before you can use RESTARTABLE_RESTORE. If you need to restart a logical restore, you must restart the database server before you can use restartable restore.

Turning on RESTARTABLE_RESTORE slows down logical restore performance. For more information, see [“Using Restartable Restore to Recover Data” on page 6-39](#). For information on the physical log, see the *Administrator’s Guide*.

Files That ON-Bar and ISM Use

Figure 9-2 lists the files that ON-Bar and ISM use and the directories in which they reside. These names and locations change if you set up the ONCONFIG file to values different than the defaults.

Figure 9-2
List of Files That ON-Bar and ISM Use

Filename	Directory	Purpose
ac_config.std	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Template for archecker parameter values. The ac_config.std file contains the default archecker (archive checking) utility parameters. To use the template, copy it into another file and modify the values.
ac_msg.log	/tmp, %INFORMIXDIR%\etc	archecker message log. When you use archecker with ON-Bar to verify a backup, it writes brief status and error messages to the ON-Bar activity log and writes detailed status and error messages to the archecker message log. Technical Support uses the archecker message log to diagnose problems with backups and restores. Specify the location of the archecker message log with the AC_MSGPATH configuration parameter.
bar_act.log	/tmp, %INFORMIXDIR%	ON-Bar activity log. For more information, see “The ON-Bar Activity Log” on page 2-17 .
bldutil.process_id	/tmp, \tmp	When the sysutils database is created, error messages appear in this file.
Emergency boot files (ixbar* files)	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Used in a cold restore. For more information, see “The Emergency Boot Files” on page 2-15 .

(1 of 3)

Filename	Directory	Purpose
ISM catalog	\$INFORMIXDIR/ism, %ISMDIR%	Records information about backup and restore save sets and storage volumes that ISM uses. ISM creates the ISM catalog during the ism_startup initialization. The ISM catalog records are stored in the mm , index , and res files. For more information, see the <i>IBM Informix Storage Manager Administrator's Guide</i> .
ISM logs	\$INFORMIXDIR /ism/logs, %ISMDIR%\logs	Operator alert messages, backend status, additional ISM information. The ISM log names are daemon.log, messages, and summary.
ISMversion	\$INFORMIXDIR/ism, %ISMDIR%\bin	Identifies the ISM version. Do not edit this file.
oncfg_servername. servernum (IDS)	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Configuration information for ON-Bar restores. The database server creates the oncfg_servername.servernum file when you initialize disk space. The database server updates the file every time that you add or delete a dbspace, a logical-log file, or a chunk. The database server uses the oncfg* file when it salvages logical-log files during a cold restore. The database server uses the oncfg* files, so do not delete them.
oncfg_servername. servernum.coserverid (XPS)	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Configuration information for ON-Bar restores. On XPS, the oncfg* files are on each coserver.
save, savegrp, savefs	\$INFORMIXDIR/bin, %ISMDIR%\bin	ISM commands use these executable files. Do not edit them.

(2 of 3)

Filename	Directory	Purpose
sm_versions	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Identifies storage manager in use. To update the storage-manager version, edit the sm_versions file directly or run the ism_startup script. For more information, see “Updating the sm_versions File” on page 3-5 .
xbsa.messages	\$INFORMIXDIR /ism/applogs, %ISMDIR%\applogs	XBSA library call information. ON-Bar and ISM use XBSA to communicate with each other. Technical Support uses the xbsa.messages log to fix problems with ON-Bar and ISM communications.
xcfg_servername.ser vernum (XPS)	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Internal server configuration information. The xcfg* file contains information about coserver location and dbslice definitions.

(3 of 3)

ON-Bar Catalog Tables

In This Chapter	10-3
The bar_action Table	10-3
The bar_instance Table	10-5
The bar_object Table	10-7
The bar_server Table	10-8
The bar_version Table	10-8
ON-Bar Catalog Map	10-9
Backup Scheduler SMI Tables	10-11
sysbuobject	10-12
sysbuobjses	10-13
sysbusession.	10-14
sysbusm	10-14
sysbusmdbspace	10-15
sysbusmlog	10-15
sysbusmworker	10-16
sysbuworker.	10-16

In This Chapter

This chapter describes the ON-Bar tables that are stored in the **sysutils** database. ON-Bar uses these tables for tracking backups and performing restores. You can query these tables for backup and restore data to evaluate performance or identify object instances for a restore.

The **bar_action** Table

The **bar_action** table lists all backup and restore actions that are attempted against an object, except during certain types of cold restores. Use the information in this table to track backup and restore history.

Column Name	Type	Explanation
act_aid	SERIAL	Action identifier. A unique number within the table. Can be used with act_oid to join with the bar_instance table.
act_oid	INTEGER	Object identifier. Identifies the backup object against which a backup or restore attempt is made. Can be used with act_aid to join with bar_instance . The act_oid column of the bar_action table equals the obj_oid column of the bar_object table.

(1 of 2)

Column Name	Type	Explanation
act_type	INTEGER	Identifies the attempted action: 1 for backup, 2 for restore, 3 for a foreign or imported restore, 4 for a fake backup, 5 for a whole-system backup, 6 for a whole-system restore, 7 for expired or deleted objects, 8 for an external restore. Fake backups, whole-system backups, and whole-system restores are available on IDS only.
act_status	INTEGER	Identifies the result of the action: 0 if successful, otherwise an ON-Bar-specific error code. For more information, see Appendix 11, "ON-Bar Messages and Return Codes."
act_start	DATETIME YEAR TO SECONDS	The date and time when the action began.
act_end	DATETIME YEAR TO SECONDS	The date and time when the action finished.

(2 of 2)

The bar_instance Table

ON-Bar writes a record to the **bar_instance** table for each successful backup. The table describes each object that is backed up. ON-Bar might later use the information for a restore operation. For example, if you specify a level-2 backup, ON-Bar uses this table to ensure that a level-1 backup was done previously.

Column Name	Type	Explanation
ins_aid	INTEGER	Action identifier. Identifies the successful action that created this instance of the backup object. Combined with ins_oid , can be used to join with the bar_action table.
ins_oid	INTEGER	Object identifier. Identifies the affected object. Can be used to join with the bar_object table. Combined with ins_aid , can be used to join with the bar_action table.
ins_time (XPS only)	INTEGER	Time stamp (real clock time). The database server uses this value when it creates the next-level backup. Value represents the number of seconds since midnight, January 1, 1970, Greenwich mean time. For IDS, the ins_time value is 0.
rsam_time	INTEGER	The backup checkpoint time stamp. Not a clock time. The database server uses this value when it creates the next level backup.
ins_level	SMALLINT	Level of the backup action: 0 for a complete backup, 1 for a backup of any changes to this object since its last level-0 backup, 2 for a backup of any changes since the last level-1 backup. This value is always 0 for logical-log backups.
ins_copyid_hi	INTEGER	The high bits of the instance copy identifier. Combined with ins_copyid_lo , it is a unique value that the storage manager assigns to link the ON-Bar object identifier with the storage-manager object identifier.

(1 of 3)

Column Name	Type	Explanation
ins_copyid_lo	INTEGER	The low bits of the instance copy identifier. Combined with ins_copyid_hi , it is a unique value that the storage manager assigns to link the ON-Bar object identifier with the storage-manager object identifier.
ins_req_aid	INTEGER	Stores the required action ID for a backup object. Used in a restore to determine which level-0 backup goes with the level-1 backup, and which level-1 backup goes with the level-2 backup. For a level-0 backup, the value of ins_req_aid is the same as ins_aid in this table. For example, if this backup is level-1, ins_req_aid holds the action ID of the corresponding level-0 backup of this object.
ins_logstream (XPS only)	INTEGER	The coserver ID from which this object came.
ins_first_log	INTEGER	In a standard backup, identifies the first logical log required to restore from this backup.
ins_chpt_log (XPS only)	INTEGER	The ID of the log that contains the rsam_time checkpoint. Used during backup to verify that logs needed for restore are backed up.
ins_last_log (XPS only)	INTEGER	Log ID of the last log needed during logical restore for this storage space to restore it to the time of the backup.
ins_partial (XPS only)	INTEGER	Value is 0 if the logical-log file was completely backed up. Value is 1 if the logical-log file was not completely backed up. Normally, log files are completely backed up, but during log salvage, the log files might be missing some data at the end. In a database server failure, the log records for the most recent transactions might not have been written to disk if you use buffered logging.
ins_sm_id (XPS only)	INTEGER	Storage-manager instance ID. Created from BAR_SM in \$ONCONFIG or %ONCONFIG%.

(2 of 3)

Column Name	Type	Explanation
ins_sm_name (XPS only)	CHAR(128)	Storage-manager instance name. Created from the BAR_SM_NAME parameter in the ONCONFIG file.
ins_verify	INTEGER	Value is 1 if the backup is verified. Value is 0 if the backup is not verified.
ins_verify_date	DATETIME YEAR TO SECOND	The current date is inserted when a backup is verified. If this backup has not been not verified, a dash represents each date and time.

(3 of 3)

The *bar_object* Table

The **bar_object** table describes each backup object. This table is a list of all storage spaces and logical logs from each database server for which at least one backup attempt was made.

Column Name	Type	Explanation
obj_srv_name	CHAR(128)	The database server name. Used to ensure that objects are restored to the correct database server. Used when multiple database servers are on the node to ensure that objects are restored in the database server instance to which the object belongs. The database server name can be up to 128 characters.
obj_oid	SERIAL	The object identifier. A unique number within the table. Can be used to join with the bar_action and bar_instance tables.

(1 of 2)

Column Name	Type	Explanation
obj_name	CHAR(128)	The user name for the object. The name can be up to 128 characters. On XPS, 15:3 is the name of log file 3 on coserver 15.
obj_type	CHAR(2)	Backup object type: CD = critical dbspace L = logical log ND = noncritical dbspace or sbpace R = rootdbs B = blobspace (IDS only)

(2 of 2)

The bar_server Table

The **bar_server** table lists the database servers in an installation. This table is used to ensure that backup objects are returned to their proper places during a restore.

Column Name	Type	Explanation
srv_name	CHAR(126)	DBSERVERNAME value specified in the ONCONFIG file. Database server name can be up to 126 characters.
srv_node	CHAR(128 or 256)	Hostname of the computer where the database server resides. Hostname can be up to 256 characters on IDS or up to 128 characters on XPS.

The bar_version Table

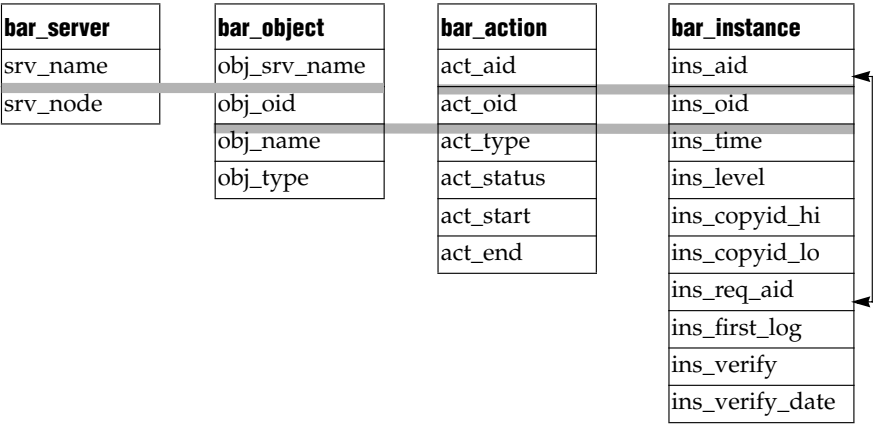
The **bar_version** table has been removed from the **sysutils** database. Use the **sm_versions** file to update the storage-manager information instead. For more information, see [“Updating the sm_versions File” on page 3-5](#).

IDS

ON-Bar Catalog Map

Figure 10-1 maps the ON-Bar tables on Dynamic Server. The gray lines show the relations between tables. The arrows show that the **ins_req_aid** value must be a valid **ins_aid** value.

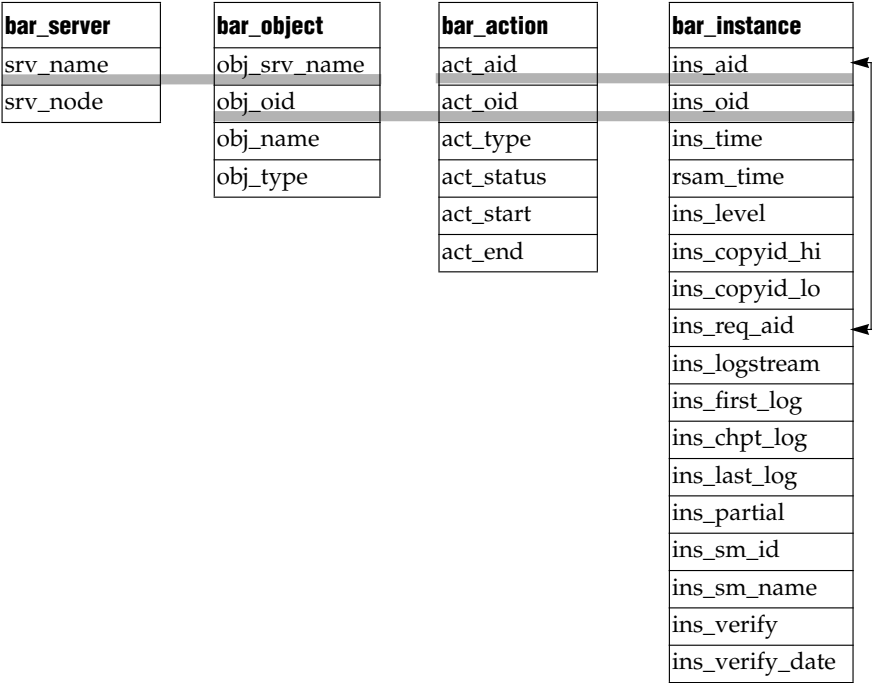
Figure 10-1
ON-Bar Catalog Map on Dynamic Server



XPS

Figure 10-2 maps the ON-Bar tables on Extended Parallel Server. The gray lines show the relations between tables. The arrows show that the **ins_req_aid** value must be a valid **ins_aid** value.

Figure 10-2
ON-Bar Catalog Map on Extended Parallel Server



Backup Scheduler SMI Tables

The following system-monitoring interface (SMI) pseudo-tables in the **sysmaster** database contain information about the Backup Scheduler. For information about other SMI tables, see the *Administrator's Reference*.

Table	Description	Reference
sysbuobject	Detailed information about backup objects that the Backup Scheduler is processing	page 10-12
sysbuobjses	Backup object name and session ID	page 10-13
sysbusession	Session status information	page 10-14
sysbusm	Storage-manager configuration information	page 10-14
sysbusmdbspace	Storage manager for backing up dbspaces on a coserver	page 10-15
sysbusmlog	Storage manager for backing up logs on a coserver	page 10-15
sysbusmworker	Coservers where onbar-worker processes can access a storage manager	page 10-16
sysbuworker	Detailed onbar-worker status	page 10-16

sysbuobject

The **sysbuobject** table provides information about the backup and restore objects that the Backup Scheduler is processing.

Column	Type	Description
owner	integer	The coserver owner of the object
priority	integer	Priority in the Backup Scheduler queue for this session
order_hi	integer	The high-order word of the restore order for the object (high 32 bits of a 64-bit number)
order_lo	integer	The low-order word of the restore order for the object (low 32 bits of a 64-bit number)
placement	integer	The storage-manager instance where this object is located
retries	integer	Number of times that the backup or restore is retried
timestamp	integer	The global time for the object
dbname	char(128)	The dbspace name
level	integer	The backup level of the dbspace
log_num	integer	The logical-log number
is_backup	integer	A value of 1 means this object is to be backed up.
is_restore	integer	A value of 1 means this object is to be restored.
is_check	integer	A value of 1 means this object is to be verified with the archecker utility.
is_block	integer	A value of 1 means the coserver is blocked for an external backup.
is_external	integer	A value of 1 means an external restore of a dbspace is in progress.
is_dbspace	integer	A value of 1 means this object is a dbspace.
is_log	integer	A value of 1 means this object is a logical-log file.

(1 of 2)

Column	Type	Description
is_coserver	integer	A value of 1 means this object is a coserver (used for external backup and restore).
is_running	integer	A value of 1 means this object is being processed.
is_ready	integer	A value of 1 means this object is ready to be processed.
is_waiting	integer	A value of 1 means this object cannot be processed yet.
is_suspend	integer	A value of 1 means processing of this object is suspended.
is_cold	integer	A value of 1 means this object is to be cold-restored.

(2 of 2)

sysbuobjses

The **sysbuobjses** table lists the backup object name and session ID. You can join the **sysbuobjses** table with the **sysbusession** table on the **ses_id** column to get all the backup objects in a specific session.

Column	Type	Description
obj_name	char(128)	Object name
ses_id	char(128)	Name of the backup or restore session

sysbusession

The **sysbusession** table shows the status of the current backup or restore session. The information in the **sysbusession** table is similar to the **onstat -g bus** output.

Column	Type	Description
id	char(128)	Name of the backup or restore session.
count	integer	Number of objects in the backup or restore session.
is_complete	integer	A value of 1 means the session is complete.
is_waiting	integer	A value of 1 means the session is waiting for work.
is_suspended	integer	A value of 1 means the session is suspended.
error	integer	Error value that is returned to the ON-Bar driver.

sysbusm

The **sysbusm** table provides information about the storage-manager configuration, storage-manager retry status, and active **onbar-worker** processes.

Column	Type	Description
id	integer	ID of this storage-manager instance.
name	char(128)	Storage-manager name.
worker	integer	Number of onbar-worker processes.
max_idle	integer	The idle time-out value. If an onbar-worker has been idle the specified time, it is terminated.
retries	integer	Times to retry the backup or restore for a storage manager.
pending	integer	Number of pending onbar-worker processes.
registered	integer	Number of registered onbar-worker processes for this storage manager.

sysbusmdbspace

The **sysbusmdbspace** table provides information about which storage manager is configured to back up dbspaces for a coserver. This table contains the value of the BAR_DBS_COSVR parameter.

Column	Type	Description
id	integer	Storage-manager ID.
cosvr_id	integer	Dbspaces on this coserver are backed up or restored to this storage manager.

sysbusmlog

The **sysbusmlog** table provides information about which storage manager is configured to back up logical logs for a coserver. This table contains the value of the BAR_LOG_COSVR parameter.

Column	Type	Description
id	integer	Storage-manager ID.
cosvr_id	integer	Logical logs on this coserver are backed up or restored to this storage manager.

sysbusmworker

The **sysbusmworker** table lists the coserver IDs of **onbar-worker** processes that are configured for a storage manager. To display the list of active **onbar-worker** processes for a storage manager, join the **sysbuworker** table with the **sysbusmworker** table (`sysbusmworker.id = sysbuworker.sm`). This table contains the same list of storage managers as the `BAR_WORKER_COSVR` parameter. For example, use the `SELECT * FROM sysbusmworker WHERE id = 1` command to display information on the storage manager listed in `BAR_SM 1`.

Column	Type	Description
id	integer	Storage-manager ID.
cosvr_id	integer	Coserver on which the onbar-worker is running.

sysbuworker

The **sysbuworker** table provides status information about **onbar-worker** processes.

Column	Type	Description
id	integer	ID of the onbar-worker .
cosvr	integer	Coserver on which the onbar-worker is running.
pid	integer	Process ID.
sm	integer	Storage-manager ID.
obj_name	char(128)	Name of the current backup or restore object.
is_new	integer	A value of 1 means this onbar-worker is initializing.
is_free	integer	A value of 1 means this onbar-worker is free for another job.
is_wait	integer	A value of 1 means this onbar-worker is waiting for its next task.

Column	Type	Description
is_busy	integer	A value of 1 means this onbar-worker is processing an object.
is_dead	integer	A value of 1 means this onbar-worker is dead.
event_dbu	integer	A value of 1 means this onbar-worker can accept dbspace backup jobs.
event_phr	integer	A value of 1 means this onbar-worker can accept dbspace restore jobs.
event_lbu	integer	A value of 1 means this onbar-worker can accept logical backup jobs.
event_lgr	integer	A value of 1 means this onbar-worker can accept logical restore jobs.
event_lbuplace	integer	A value of 1 means this onbar-worker determines the correct storage manager to handle logical backups.
event_phrplace	integer	A value of 1 means this onbar-worker determines the correct storage manager to handle dbspace restores.
event_dbplace	integer	A value of 1 means this onbar-worker determines the correct storage manager to handle dbspace backups.
event_lgrplace	integer	A value of 1 means this onbar-worker determines the correct storage manager to handle logical restores.
event_cold	integer	A value of 1 means this onbar-worker can process an object that is part of a cold restore.
idle_time	integer	How long this onbar-worker has been idle.

(2 of 2)

ON-Bar Messages and Return Codes

In This Chapter	11-3
About ON-Bar Messages	11-3
Message Format	11-3
Message Numbers.	11-4
ON-Bar Usage Messages	11-5
ON-Bar Return Codes	11-9

In This Chapter

The first part of this chapter describes the ON-Bar activity log file and the ON-Bar usage messages. Both Dynamic Server, Version 9.4, and Extended Parallel Server, Version 8.40, share common ON-Bar messages.

The second part of this chapter describes the ON-Bar return codes.

For information on ON-Bar informational, progress, warning, and error messages, use the **finderr** or **Error Messages** utility or view *IBM Informix Error Messages* at

<http://www.ibm.com/software/data/informix/pubs/library/>.

About ON-Bar Messages

This section explains how to read and interpret messages in the ON-Bar activity log file. For more information, see [“The ON-Bar Activity Log” on page 2-17](#).

Message Format

A message in the ON-Bar activity log file has the following format:

```
timestamp process_id parent_process_id message
```

Figure 11-1 describes each field in the message. No error message numbers appear in the ON-Bar activity log.

Figure 11-1
ON-Bar Message Format

Message Field	Description
<i>timestamp</i>	Date and time when ON-Bar writes the message.
<i>process id</i>	The number that the operating system uses to identify this instance of ON-Bar.
<i>parent process id</i>	The number that the operating system uses to identify the process that executed this instance of ON-Bar.
<i>message</i>	The ON-Bar message text.

The following example illustrates a typical entry in the ON-Bar activity log:

```
1999-08-18 10:09:59 773 772 Completed logical restore.
```

Important: If you receive an XBSA error message, consult the storage-manager logs for more details.



Message Numbers

The ON-Bar message numbers range from -43000 to -43421. The following table lists the ON-Bar message groups. Because message numbers do not display in the activity log, the best way to find information about ON-Bar messages is to search the message text in *IBM Informix Error Messages* at <http://www.ibm.com/software/data/informix/pubs/library/>.

ON-Bar Message Type	Message Numbers
ON-Bar usage	-43000 to -43009
Options checking	-43010 to -43034
Permission checking	-43035 to -43039
Emergency boot file interface	-43040 to -43059
ONCONFIG file interface	-43060 to -43074
Operating system interface	-43075 to -43099

(1 of 2)

ON-Bar Message Type	Message Numbers
Database server interface	-43100 to -43229
Backup and restore status	-43230 to -43239
Onbar-worker processes	-43240 to -43254
XBSA interface	-43255 to -43304
onsmsync	-43305 to -43319
archecker	-43320 to -43334
ondblog	-43400 to -43421

(2 of 2)

ON-Bar Usage Messages

This section lists usage messages only. All informational, progress, warning, and error messages appear in **finderr** and *IBM Informix Error Messages* at <http://www.ibm.com/software/data/informix/pubs/library/>.



Important: You must specify the *-b*, *-v*, or *-r* option first in the command so that ON-Bar can determine whether it is performing a backup, verification, or restore. The only exception is when you start or stop the ON-Bar session command on Extended Parallel Server.

-43000

ON-Bar backup and verification usage.

IDS

```
For Dynamic Server
-b [-L level] [-w | -f filename | spaces] [-O]
-b -F
-b -l [-c | -C | -s] [-O]
-v [-p] [-t time] [-f filename | spaces]

    -b back up
    -c back up current logical log
    -C continuous logical-log backup
    -f pathname of file containing list of storage spaces
    -F fake backup
    -l back up full logical logs (no spaces)
    -L back up level: 0, 1, or 2; defaults to 0
    -O override internal error checks - enforce policy strictly
    -w whole-system backup
    -s salvage logs
    -v verify consistency of most recent backups
    spaces list of storage spaces
```



XPS

```
For Extended Parallel Server
-b [-v] [-p] [-q name] [-L level] [-f filename | spaces]
-b -l [-q name] [-s] [-f filename | coservers]
-v [-p] [-t time] [-q name] [-L level] [-f filename | spaces]

    -b back up
    -f pathname of file containing list of storage spaces
    -l back up full logical logs (no spaces)
    -L back up level: 0, 1, or 2; defaults to 0
    -p back up spaces only (no logs)
    -q identifies the backup session, default DBSERVERNAMEpid
    -s salvage logs
    -v verify consistency of most recent backups
        (For details, contact Technical Support)
    spaces list of storage spaces
    coservers coserverlogs to back up
```



The backup or verification command was entered incorrectly. Revise the command and try again. The **-b** or **-v** parameter must come first in the command.

-43001

ON-Bar restore usage.

IDS

For Dynamic Server

```
-r [-e] [-O] [-f filename | spaces]
-r [-e] [-t time | -n log] [-O]
-r -p [-e] [-t time] [-O] [-f filename | spaces]
-r -l [-t time | -n log]
-r -w [-e] [[-p] [-t time] | -n log] [-O]
-RESTART
-r -rename -f filename [-w] [-p] [-t time] [-n log] [-f filename | spaces]
-r {-rename -p old path -o old offset -n new path -o new offset...}
  [-w] [-p] [-t time] [-n log] [-f filename | spaces]
```

◆

XPS

For Extended Parallel Server

```
-r [-e] [-q name] [-O] [-f filename | spaces]
-r [-e] [-q name] [-t time] [-O]
-r -p [-e] [-q name] [-t time] [-O] [-f filename | spaces]
-r -l [-q name] [-t time] [-f filename | logstreams]

-e external restore
-f pathname of file containing list of storage spaces
-l logical-log only restore (no spaces)
-n last logical log to restore
-O override internal error checks - enforce policy strictly
-p physical only restore (no logs)
-q name to identify the restore session
-r restore
-t point in time to stop restore
-w whole system to restore
-RESTART restart an interrupted restore
spaces list of storage spaces or logstreams to restore
With -rename option:
-rename -f filename pathname of file containing list of mapped
        chunk pathnames and offsets
-p old pathname of chunk
-o old offset of chunk
-n new pathname of chunk
-o new offset of chunk
```

Renaming chunks during restore is not allowed for HDR setup.

◆

The restore command was entered incorrectly. Revise the command and try again. You must specify the **-r** parameter first.

-43002

XPS

ON-Bar session usage.

```
For Extended Parallel Server
{on | ON | off | OFF | -d} [-q] session_name

on/ON      resume a suspended backup/restore session
off/OFF    suspend a backup/restore session
-d         destroy backup/restore session
-q         name to identify the session
```

The session command was entered incorrectly. Revise the command and try again. The session name is required. ♦

-43003

XPS

onbar_w usage.

```
For Extended Parallel Server
[-b] [-d] [-l] [-r] [-p]

-b accept backup requests
-d accept db/blobspaces
-l accept logical logs
-r accept restore requests
-p accept storage manager placement requests
If no commands are entered, accept all requests by default.
```

The **onbar_w** command was entered incorrectly. Revise the command and try again. ♦

-43006

onsmsync usage.

```
onsmsync [-g gen | -t time | -i interval] [-O]
          [-f filename | spaces]

onsmsync -b

-b just regenerate the emergency boot file
-f pathname of file containing list of storage spaces
-g number of generations/versions of level-0 backup to
  retain
-i time interval (age) before which objects should be
  expired
-t datetime before which objects should be expired
-O override internal error checks - enforce policy strictly
spaces list of storage spaces to check for expiration
```

The **onsmsync** command was entered incorrectly. Revise the command and try again.

ON-Bar Return Codes

Figure 11-2 shows the ON-Bar return codes for all Informix database servers. These return codes are accompanied by messages in the ON-Bar activity log. For details about an ON-Bar or storage-manager error, review the activity log before you call Technical Support.

Figure 11-2
Common ON-Bar Return Codes

Decimal Value	ON-Bar Return Code Description
2 through 34	These return codes are produced by XBSA. For more information, consult your storage-manager documentation and log files.
100	ON-Bar cannot find something in sysutils , the emergency boot file, or storage-manager catalogs that it needs for processing. Check the ON-Bar activity log for messages that say what could not be found and try to resolve that problem. If the problem recurs, contact Technical Support.
104	Adstar Distributed Storage Manager (ADSM) is in generate-password mode. ON-Bar does not support ADSM running in generate-password mode. For information on changing the ADSM security configuration, refer to your ADSM manual.
119	The logical log is full on one or more coservers. Perform a logical-log backup.
120	The transport buffer size has changed since this object was last backed up. This object cannot be restored. Set the transport-buffer size to its original value and retry the restore.
121	Error occurred on dbslice name expansion (XPS). ON-Bar was unable to determine the list of dbspaces in a dbslice.
122	Deadlock detected. The ON-Bar command is contending with another process. Retry the ON-Bar command.

(1 of 7)

Decimal Value	ON-Bar Return Code Description
123	<p>The root dbspace was not in the cold restore.</p> <p>You cannot perform a cold restore without restoring the root dbspace. To resolve the problem, try one of the following procedures:</p> <ul style="list-style-type: none"> ■ Bring the database server to quiescent or online mode and restore just the storage spaces that need to be restored. ■ If the database server is offline, issue the onbar -r command to restore all the storage spaces. ■ Make sure that the root dbspace and other critical dbspaces are listed on the command line or in the -f filename.
124	<p>The buffer had an incomplete page during the backup.</p> <p>For assistance, contact Technical Support.</p>
126	<p>Error processing the emergency boot file.</p> <p>Check the ON-Bar activity log for descriptions of the problem and the emergency boot file for corruption such as non-ASCII characters or lines with varying numbers of columns. If the source of the problem is not obvious, contact Technical Support.</p>
127	<p>Could not write to the emergency boot file.</p> <p>Often, an operating-system error message accompanies this problem. Check the permissions on the following files and directories:</p> <ul style="list-style-type: none"> ■ \$INFORMIXDIR/etc on UNIX or %INFORMIXDIR%\etc on Windows ■ The emergency boot file ■ The directory that BAR_BOOT_DIR points to (XPS only).
128	<p>Data is missing in the object description.</p> <p>For assistance, contact Technical Support.</p>
129	<p>ON-Bar received a different object for restore than it had expected. (The backup object did not match.) The requested backup object might have been deleted or expired from the storage manager.</p> <p>Run onsmsync to synchronize the sysutils database, emergency boot file, and storage-manager catalogs. For assistance, contact Technical Support.</p>

(2 of 7)

Decimal Value	ON-Bar Return Code Description
130	<p>Database server is not responding.</p> <p>The database server probably failed during the backup or restore. Run the onstat - command to check the database server status and then:</p> <ul style="list-style-type: none"> ■ If the operation was a cold restore, restart it. ■ If the operation was a backup or warm restore, restart the database server and retry the backup or warm restore.
131	<p>A failure occurred in the interface between ON-Bar and the database server.</p> <p>For assistance, contact Technical Support.</p>
132	<p>Function is not in the XBSA shared library.</p> <p>Verify that you are using the correct XBSA for the storage manager. For information, consult your storage-manager manual.</p>
133	<p>Failed to load the XBSA library functions.</p> <p>Verify that you are using the correct XBSA for the storage manager. Ensure that the <code>BAR_BSALIB_PATH</code> value in the <code>ONCONFIG</code> file points to the correct location of the XBSA shared library. For information, consult your storage-manager manual.</p>
134	<p>User wants to restore a logical-log file that is too early.</p> <p>You probably tried a point-in-log restore (onbar -r -l -n) after performing a separate physical restore. The specified logical log is too old to match the backups used in the physical restore. Perform either of the following steps:</p> <ul style="list-style-type: none"> ■ Rerun the physical restore from an older set of physical backups. ■ Specify a later logical log in the -n option when you rerun the point-in-log restore. To find the earliest logical log that you can use, look at the emergency boot file. For assistance, contact Technical Support.
136	<p>ON-Bar cannot warm restore the critical dbspaces.</p> <p>Perform either of the following steps:</p> <ul style="list-style-type: none"> ■ Reissue the warm-restore command without listing any critical dbspaces. ■ Shut down the database server and perform a cold restore.
137	<p>The <code>MAX_DBSPACE_COUNT</code> was exceeded.</p> <p>For assistance, contact Technical Support.</p>

(3 of 7)

Decimal Value	ON-Bar Return Code Description
138	<p>An XBSA error occurred.</p> <p>Verify that you are using the correct XBSA for the storage manager. Also check the bar_act.log for XBSA error messages. For information, consult your storage-manager manual.</p>
139	<p>Either the XBSA version is missing from the sm_versions file or the incorrect XBSA version is in the sm_versions file.</p> <p>Insert the correct XBSA version into the sm_versions file. For more information, consult your storage-manager manual.</p>
140	<p>A fake backup failed.</p> <p>Retry the fake backup using the onbar -b -F command. Only IDS supports fake backups. If the fake backup fails again, contact Technical Support.</p>
141	<p>ON-Bar received an operating-system signal. Most likely, the user entered the Ctrl-C command to stop an ON-Bar process.</p> <p>Fix the cause of the interruption and then retry the ON-Bar command.</p>
142	<p>ON-Bar was unable to open a file.</p> <p>Verify that the named file exists and that the permissions are correct. Check the ON-Bar activity log for an operating-system error message.</p>
143	<p>ON-Bar was unable to create a child process.</p> <p>If BAR_MAX_BACKUP is not 0, ON-Bar could not create child processes to perform the parallel backup or restore. The operating system probably ran out of resources. Either not enough memory is available to start a new process or no empty slot exists in the process table.</p> <p>Check the operating-system logs, the ON-Bar activity log, or the console.</p>
144	<p>The log backup was aborted because one or more blobspaces were down.</p> <p>Attempt to restore the blobspace. If the restore fails, retry the log backup using the onbar -l -O command. Executing this command might make the blobspace unrestorable.</p>
145	<p>ON-Bar was unable to acquire more memory space.</p> <p>Wait for system resources to free up and retry the ON-Bar command.</p>
146	<p>ON-Bar was unable to connect to the database server.</p> <p>The network or the database server might be down. For assistance, contact Technical Support.</p>

Decimal Value	ON-Bar Return Code Description
147	<p>ON-Bar was unable to discover any storage spaces or logical logs to back up or restore.</p> <p>For example, if you specify a point-in-time restore but use a <i>datetime</i> value from before the first standard backup, ON-Bar cannot build a list of storage spaces to restore. This return code also displays if you specify a whole-system restore without having performed a whole-system backup.</p> <p>Verify that the database server and the storage spaces are in the correct state for the backup or restore request. Contact Technical Support.</p>
148	<p>An internal SQL error occurred.</p> <p>Provide Technical Support with the information from the ON-Bar activity log.</p>
149	<p>Either you entered the wrong ON-Bar syntax on the command line or entered an invalid or incorrect <i>datetime</i> value for your GLS environment.</p> <p>Check the command that you tried against the usage message in the ON-Bar activity log. If that does not help, then retry the command with quotes around the <i>datetime</i> value. If your database locale is not English, use the GL_DATE or GL_DATETIME environment variables to set the date and time format.</p>
150	<p>Error collecting data from the ONCONFIG file.</p> <p>Check the permissions, format, and values in the ONCONFIG file. Check that the ONCONFIG environment variable is set correctly.</p>
151	<p>The database server is in an incorrect state for this backup or restore request, or an error occurred while determining the database server state.</p> <p>Either you attempted an operation that is not compatible with the database server mode or ON-Bar is unable to determine the database server state. For example, you cannot do a physical backup with the database server in recovery mode.</p> <p>Check the error message in the ON-Bar activity log. If an ASF error occurred, the following message displays in the ON-Bar activity log:</p> <pre>Fatal error initializing ASF; asfcode = code</pre> <p>To determine the cause of the ASF error, refer to the ASF error code in this message and repeat the backup or restore command. If an ASF error did not occur, change the database server state and repeat the backup or restore command.</p>

(5 of 7)

Decimal Value	ON-Bar Return Code Description
152	<p>ON-Bar cannot back up the logical logs.</p> <p>The logical logs are not backed up for either of the following reasons:</p> <ul style="list-style-type: none"> ■ If another log backup is currently running. ■ If you perform a logical-log backup with the LTAPEDEV parameter set to /dev/null (UNIX) or null (Windows). <p>You receive this return code when no log backups can be done.</p> <p>To enable log backups, change the LTAPEDEV parameter to a valid value.</p>
153	<p>ON-Bar cannot set the process group id. If BAR_MAX_BACKUP is set to any value other than 1 and ON-Bar encounters an error setting the process group id, this value is returned.</p> <p>This message is a warning of a possible operating-system problem.</p>
154	<p>The ON-Bar user does not have the correct permissions.</p> <p>You must be user root or informix or a member of the bargroup group on UNIX or a member of the Informix-Admin group on Windows to execute ON-Bar commands.</p>
155	<p>The INFORMIXSERVER environment variable is not set.</p> <p>Set the INFORMIXSERVER environment variable to the correct database server name.</p>
157	<p>Error attempting to set the INFORMIXSHMBASE environment variable to -1.</p> <p>ON-Bar could not set INFORMIXSHMBASE to -1. For assistance, contact either the system administrator or Technical Support.</p>
158	<p>An internal ON-Bar error occurred.</p> <p>Contact Technical Support.</p>
159	<p>An unexpected error occurred.</p> <p>Contact Technical Support.</p>
160	<p>External restore failed.</p> <p>To determine what went wrong with the external restore, look at the bar_act.log and the online.log files. Ensure that you already performed the manual part of the external restore before you retry the onbar-r -e command to complete the external restore. If that does not work, try the external restore from a different external backup.</p>
161	<p>Restarted restore failed.</p> <p>Verify that RESTARTABLE_RESTORE is set to ON and try the original restore again. For more information, check the ON-Bar activity log and database server message logs (IDS).</p>

Decimal Value	ON-Bar Return Code Description
177	An online dbspace was restored. This return code notifies the user that the -O option overrode the internal checks in ON-Bar. You do not need to take any action.
178	The logical log was backed up while one or more blobspaces were down. This return code notifies the user that the -O option overrode the internal checks in ON-Bar. Examine the data in the blobspace to determine which simple large objects you need to re-create. These blobspaces might not be restorable. For assistance, contact Technical Support.
179	ON-Bar created the chunk needed to restore the dbspace. This return code notifies the user that the -O option overrode the internal checks in ON-Bar. You do not need to take any action.
180	ON-Bar could not create the chunk needed to restore the dbspace. Create the chunk file manually. Retry the restore without the -O option.
181	ON-Bar expired an object that was needed for a backup or restore. The onmsmsync utility expired an object that might be needed for a restore. You probably specified onmsmsync with the -O option. If you used the -O option by mistake, contact Technical Support to recover the object from the storage manager.
247	Merging of emergency boot files timed out because it took too long. (XPS) On UNIX, look in /tmp/bar_act.log and the file that the BAR_ACT_LOG parameter points to for clues. (The onbar-merger writes to /tmp/bar_act.log until it has enough information to read the ONCONFIG file.) Resolve the problems that the bar_act.log describes and retry the cold restore. If the cold restore still fails, contact Technical Support.
252	Received the wrong message from onbar_m that merges the emergency boot files. (XPS) For assistance, contact Technical Support.

(7 of 7)

The ontape Backup and Restore System

- Chapter 12** **Configuring ontape**
- Chapter 13** **Backing Up with ontape**
- Chapter 14** **Restoring with ontape**

Section III



Configuring ontape

In This Chapter	12-3
Setting the ontape Configuration Parameters	12-3
Setting the Tape-Device Parameters	12-4
Specifying Separate Devices for Storage-Space and Logical-Log Backups	12-5
Using Symbolic Links to Specify Tape Devices.	12-6
Specifying a Remote Device	12-6
Using /dev/null for a Tape Device	12-6
Rewinding Tape Devices Before Opening and on Closing	12-7
Specifying the Tape-Block-Size Parameters	12-7
Specifying the Tape-Size Parameters	12-7
Checking and Changing ontape Configuration Parameters	12-8
Who Can Change ontape Parameters?	12-8
When Can You Change ontape Parameters?	12-8
Changing TAPEDEV to /dev/null.	12-9
Changing LTAPEDEV to /dev/null	12-9
Verifying That the Tape Device Can Read the Specified Block Size.	12-9
Changing ontape Parameters	12-10
Changing Backup-Tape Parameters	12-10
Changing Logical-Log Backup Tape Parameters	12-10

In This Chapter

This chapter explains how to set the configuration parameters that the **ontape** utility uses for backups of storage spaces and logical logs. For a description of how **ontape** differs from ON-Bar, see [“Comparing ON-Bar and ontape” on page 1-12](#).

This chapter describes the following tasks:

- Setting the configuration parameters for **ontape**
- Checking configuration parameters for **ontape**
- Changing configuration parameters for **ontape**

[Chapter 13, “Backing Up with ontape”](#) describes how to use the **ontape** utility to back up storage spaces and logical-log files.

Setting the ontape Configuration Parameters

The **ontape** utility uses six parameters in the ONCONFIG file to create storage-space and logical-log backups. The ONCONFIG file is located in the `$INFORMIXDIR/etc` directory. You specify that file in the **ONCONFIG** environment variable. For a description of the **ONCONFIG** environment variable and instructions on how to set it, see the *IBM Informix Guide to SQL: Reference*.

The first set of ONCONFIG parameters specifies the characteristics of the tape device and tapes for storage-space backups; the second set specifies the characteristics of the tape device and tapes for logical-log backups.

The following list shows backup tape devices and their associated tape parameters.

TAPEDEV	is the tape device used for storage-space backups.
TAPEBLK	is the block size of the tapes used for storage-space backups, in kilobytes.
TAPESIZE	is the size of the tapes used for storage-space backups, 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 storage-space 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.

Specifying Separate Devices for Storage-Space and Logical-Log Backups

When possible, the LTAPEDEV and TAPEDEV parameters in the ONCONFIG file must each specify a different device. When you specify separate devices for storage-space and logical-log backups, you can schedule these backups independently of each other. You can create a backup 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 the database server to stop processing during a backup. When this happens, you face limited options. You can either abort the backup to free the tape device and back up the logical-log files or leave normal processing suspended until the backup completes.

Precautions to Take When You Use One Tape Device

When only one tape device exists and you want to create backups while the database server is online, take the following precautions:

- Configure the database server with a large amount of logical-log space through a combination of many or large logical-log files. (See your *Administrator's Guide*.)
- Store all explicitly created temporary tables in a dedicated dbspace and then drop the dbspace before backing up.
- Create the backup when low database activity occurs.
- Free as many logical-log files as possible before you begin the backup.

The logical log can fill up before the backup completes. The backup synchronizes with a checkpoint. A backup might wait for a checkpoint to synchronize activity, but the checkpoint cannot occur until all virtual processors exit critical sections. When database server processing suspends because of a full logical-log file, the virtual processors cannot exit their critical sections and a deadlock results.

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
```

When you set the LTAPEDEV configuration parameter, as the following example shows, you can switch to a different device without changing the LTAPEDEV parameter:

```
LTAPEDEV /dbfiles/logtape
```

You only need to change the symbolic link, as the following example shows:

```
ln -s /usr/backups /dbfiles/logtape
```

A user with one tape device could redirect a logical-log backup to a disk file while using the tape device for a backup.

Specifying a Remote Device

You can perform a storage-space or logical-log backup across your network to a remote device attached to another host computer. You should 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
```

For information on the tape size for remote devices, see [“Tape Size for Remote Devices” on page 12-8](#).

Using /dev/null for a Tape Device

It is recommended that you do not use **/dev/null** as the device when backing up. However, when you specify **/dev/null** as a backup tape device, you can avoid the overhead of a level-0 backup that is required after some operations, such as changing the logging status of a database. Obviously, you cannot restore storage spaces from a backup to **/dev/null**.



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**, the database server marks the logical-log files as backed up as soon as they become full, effectively discarding logical-log information.

Rewinding Tape Devices Before Opening and on Closing

With **ontape**, you must use *rewindable* tape devices. Before reading from or writing to a tape, the database server performs a series of checks that require the rewind.

Specifying the Tape-Block-Size Parameters

Specify the block-size parameters `TAPEBLK` and `LTAPEBLK` 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.

The **ontape** utility 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. `TAPESIZE` and `LTAPESIZE` specify the maximum amount of data that you can write to a tape.

To write or read the tape to the end of the device, set `TAPESIZE` and `LTAPESIZE` to 0.

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 to a remote device, the amount of data written to the tape is the smaller of LTAPESIZE and the following formula:

$$(\text{sum of space occupied by all logical-log files on disk}) - (\text{largest logical-log file})$$

The I/O to the remote device completes and the database server frees the logical-log files before a log-full condition occurs.

Checking and Changing ontape Configuration Parameters

To examine your ONCONFIG file (the file specified in \$INFORMIXDIR/etc/\$ONCONFIG), use one of the following:

- Execute **onstat -c** while the database server is running.
- Use IBM Informix Server Administrator (**Configuration**→**ONCONFIG**).

This section provides information on how to change configuration parameters for **ontape**.

Who Can Change ontape Parameters?

When you log in as either user **informix** or **root**, you can use a text editor or ISA to change the value of configuration parameters for **ontape**.

When Can You Change ontape Parameters?

You can change the values of parameters for **ontape** while the database server is online. The change takes effect immediately. 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 a backup, the database server bypasses the backup but still updates the dbspaces with the new backup time stamps. When you set TAPEDEV to **/dev/null**, you must do it before you start **ontape** to request the backup. No problems exist when you change TAPEDEV to **/dev/null** while the database server is online and **ontape** is not running.

Changing LTAPEDEV to /dev/null

Take the database server offline before you change the value of LTAPEDEV to **/dev/null**. When you make the change while the database server is either quiescent or online, you can create a situation where you back up one or more logical-log files but do not free them. This situation can interrupt processing because the database server stops when it finds that the next logical-log file (in sequence) is not free.

When you set LTAPEDEV to **/dev/null**, the 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 database server can reuse them.

Verifying That the Tape Device Can Read the Specified Block Size

The **ontape** utility 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

Before you change the parameters for **ontape**, perform a level-0 backup, as explained in [“Performing a Backup” on page 13-10](#).

Changing Backup-Tape Parameters

To change the value of TAPEDEV, TAPEBLK, and TAPESIZE from the command line, use a text editor to edit your ONCONFIG file. Save the file. The change takes effect immediately. You also can change these parameters in ISA.

Changing Logical-Log Backup Tape Parameters

To change the value of LTAPEDEV, LTAPEBLK, and LTAPESIZE from the command line, use a text editor to edit your ONCONFIG file. Save the file. The change takes effect immediately. You also can change these parameters in ISA.

Backing Up with ontape

In This Chapter	13-3
Syntax of ontape	13-3
Starting ontape	13-4
Using ontape Exit Codes	13-4
Changing Database Logging Status	13-5
Creating a Backup	13-6
Choosing a Backup Level	13-6
Level-0 Backups	13-6
Level-1 Backups	13-6
Level-2 Backups	13-6
Backing Up After Changing the Physical Schema	13-7
Preparing for a Backup	13-7
Avoid Using Temp Tables During Heavy Activity	13-8
Making Sure Enough Logical-Log Space Exists	13-8
Keeping a Copy of Your ONCONFIG File	13-8
Verifying Consistency Before a Level-0 Backup	13-8
Using Online and Quiescent Backups	13-9
Ensuring That the Operator Is Available	13-9
Labelling Tapes Created with ontape	13-10
Performing a Backup	13-10
Backup Examples	13-11
Backing Up Raw Tables	13-12
When the Logical-Log Files Fill During a Backup	13-12
When You Can Use Two Tape Devices	13-12
When Only One Tape Device Is Available	13-12
When a Backup Terminates Prematurely	13-13
Monitoring Backup History Using oncheck	13-13

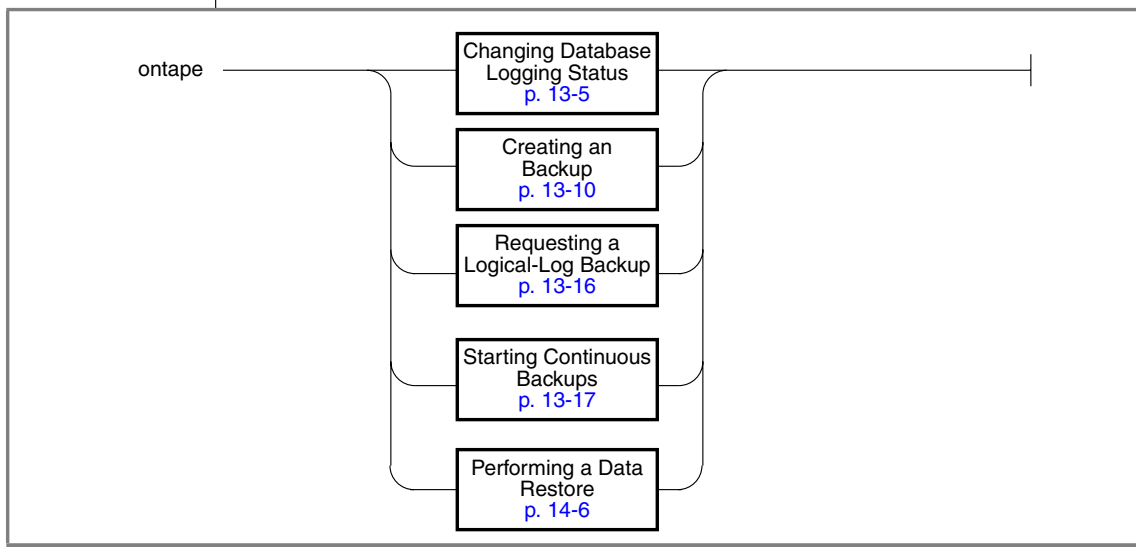
Backing Up Logical-Log Files with ontape	13-14
Before You Back Up the Logical-Log Files	13-14
Using BlobSpace TEXT and BYTE Data Types and	
Logical-Log Files	13-14
Using /dev/null When You Do Not Need to Recover	13-15
When Must You Back Up Logical-Log Files?	13-15
Starting an Automatic Logical-Log Backup	13-16
Starting a Continuous Logical-Log File Backup	13-16
Ending a Continuous Logical-Log Backup	13-17
What Device Must Logical-Log Backups Use?	13-18

In This Chapter

This chapter describes how to use **ontape** to back up storage spaces and logical-log files, and how to change the database logging status. The **ontape** utility can back up and restore the largest chunk files that your database server supports. The **ontape** utility cannot back up temporary dbspaces and temporary sbspaces.

Syntax of ontape

The **ontape** utility enables you to change the logging status of a database, back up storage spaces and logical-log files, perform continuous logical-log backups, and restore data from a backup. The following syntax diagram illustrates the basic syntax of the **ontape** utility.



Starting *ontape*

When you need more than one tape during a backup, **ontape** prompts for each additional tape.

If the database server is in maintenance mode, for example, during a conversion, then *ontape* can only be started by one of the following users:

- **root**
- **informix**
- The user who started the database server (if not the user **root** or **informix**)



Warning: Do not start *ontape* in background mode (that is, using the UNIX **&** operator on the command line). 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.

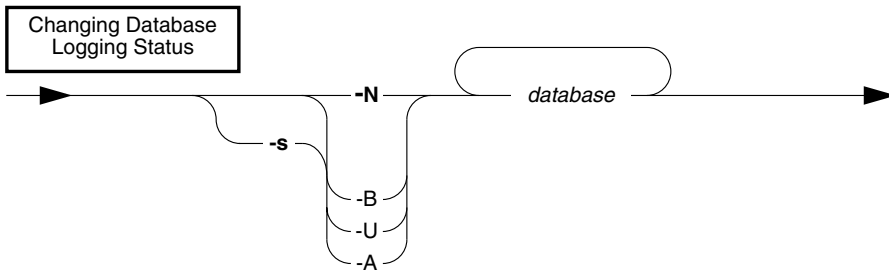
The *ontape* utility does not include default values for user interaction, nor does it support retries. When *ontape* expects a yes/no response, it assumes that any response not recognized as a “yes” is “no”.

Using *ontape* 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 backup 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 a backup.

-U directs **ontape** to change the status of the specified database to unbuffered logging.

For considerations about changing the logging status of a database, see your *Administrator's Guide*.

Creating a Backup

This section explains how to plan for and create backups of your database server data.

Choosing a Backup Level

The **ontape** utility supports level-0, level-1, and level-2 backups. For information on scheduling backups, see [“Planning a Recovery Strategy” on page 1-14](#).



Tip: *It is good practice to create a backup schedule that keeps level-1 and level-2 backups small and to schedule frequent level-0 backups. With such a backup schedule, you avoid having to restore large level-1 and level-2 backups or many logical-log backups.*

Level-0 Backups

When a fire or flood, for example, completely destroys a computer, you need a level-0 backup to completely restore database server data on the replacement computer. For online backups, the data on the backup tape reflects the contents of the storage spaces at the time the level-0 backup began. (The time the backup started could reflect the last checkpoint before the backup started.)

A level-0 backup can consume lots of time because **ontape** must write all the pages to tape.

Level-1 Backups

A level-1 backup usually takes less time than a level-0 backup because you copy only part of the database server data to the backup tape.

Level-2 Backups

A level-2 backup after a level-1 backup usually takes less time than another level-1 backup because only the changes made after the last level-1 backup (instead of the last level-0) get copied to the backup tape.

Backing Up After Changing the Physical Schema

You must perform a level-0 backup to ensure that you can restore the data after you make the following administrative changes. Consider waiting to make these changes until your next regularly scheduled level-0 backup.

- Changing TAPEDEV or LTAPEDEV from `/dev/null`
- Adding logging to a database
- Adding a dbspace, blobspace, or sbpace before you can restore it with anything less than a full-system restore
- Starting mirroring for a dbspace that contains logical-log files
- Dropping a logical-log file
- Moving one or more logical-log files
- Changing the size or location of the physical log and after you reinitialize shared memory
- Dropping a chunk before you can reuse the dbspace that contains that chunk
- Renaming a chunk during a cold restore



Tip: Although you no longer need to backup immediately after adding a logical-log file, your next backup should be level-0 because the data structures have changed.

Preparing for a Backup

When you create a backup, take the following precautions:

- Avoid using temp tables during heavy activity.
- Make sure you make sufficient logical-log space to create a backup.
- Keep a copy of your ONCONFIG file.
- Verify data consistency.
- Run the database 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 a backup while using the **ontape** utility, that table is placed in DBSPACETEMP. When heavy activity occurs during the backup process, the temp table can keep growing and can eventually fill up DBSPACETEMP. When this situation occurs, the backup aborts and your monitor displays a NO FREE DISK error message.

Making Sure Enough Logical-Log Space Exists

When the total available space in the logical log amounts to less than half a single logical-log file, the database server does not create a backup. You must back up the logical-log files and attempt the backup again.

You cannot add mirroring during a backup.



Important: *When you use only one available tape device, make sure you back up all your logical-log files before you start your backup to reduce the likelihood of filling the logical log during the backup.*

Keeping a Copy of Your ONCONFIG File

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

Verifying Consistency Before a Level-0 Backup

To ensure the integrity of your backups, periodically verify that all database server data and overhead information is consistent before you create a full-system level-0 backup. You need not check this information before every level-0 backup, but we recommend that you keep the necessary tapes from the most recent backup created immediately after the database server was verified as consistent. For information on consistency checking, see your *Administrator's Guide*.

Using Online and Quiescent Backups

You can create a backup while the database server is online or in quiescent mode. The terminal you use to initiate the backup command is dedicated to the backup (displaying messages) until the backup completes. Once you start a backup, the database server must remain in the same mode until the backup finishes; changing the mode terminates the backup activity.

What Is an Online Backup?

You can use an online backup when you want your database server accessible while you create the backup.

Some minor inconveniences can occur during online backups. An online backup 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 the database server during a backup.

During an online backup, allocation of some disk pages in storage spaces can temporarily freeze. Disk-page allocation is blocked for one chunk at a time until you back up the used pages in the chunk.

What Is a Quiescent Backup?

You create a quiescent backup while the database server is quiescent. Use quiescent backups when you want to eliminate partial transactions in a backup.

Do not use quiescent backups when users need continuous access to the databases.

Ensuring That the Operator Is Available

Keep an operator available during a backup to mount tapes as prompted.

A backup could take several reels of tape. When an operator is not available to mount a new tape when one becomes full, the backup waits. During this wait, when the backup is an online backup, the physical log space could fill up, and that causes the database server to abort the backup. Thus, make sure an operator is available.

Labelling Tapes Created with `ontape`

When you label tapes created with **ontape**, the label must include the following information:

- Backup level
- Date and time
- Tape number that **ontape** provides

The following example shows what a label can look like:

```
Level 1: Wed Nov 27, 2001 20:45 Tape # 3 of 5
```

Each backup begins with its first tape reel numbered 1. You number each additional tape reel consecutively thereafter. You number a five-tape backup 1 through 5. (Of course, it is possible that you could not know that it is a five-tape backup until it is finished.)

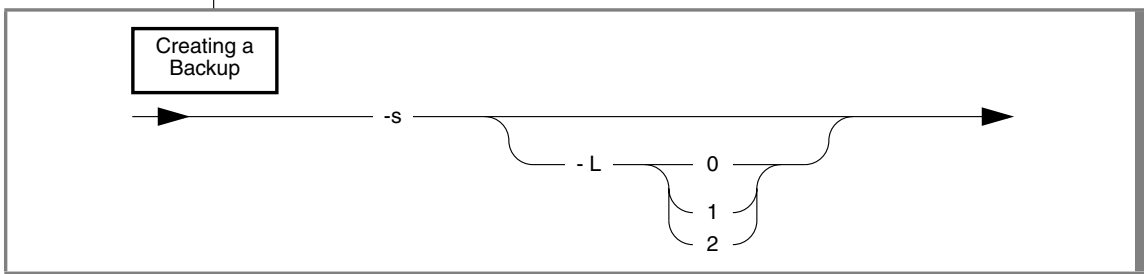
Performing a Backup

Before you begin a backup, perform the following steps:

- Place a write-enabled tape on the tape-drive device that `TAPEDEV` specifies.
- Put the device online with the appropriate operating-system command.
- Place the database server in online or quiescent mode

Do not store more than one backup on the same tape; begin every backup with a different tape. (Often, a backup spans more than one tape.)

To create a backup, use the **-s** option of the **ontape** command.



- s directs **ontape** to create a backup. Also prompts you to supply the backup level (0, 1, or 2) that you want to create.
- L directs **ontape** to create a backup of the level specified. When you use the **-L** option to specify the backup level as part of the command, you can avoid being prompted for it.

The **ontape** utility backs up the storage spaces in the following order: root dbspaces, blobspaces, sbspaces, and dbspaces.

A backup 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 the backup is complete.

Backup Examples

Execute the following command to start a backup without specifying a level:

```
ontape -s
```

You can use the **-L** option to specify the level of the backup as part of the command, as the following example shows:

```
ontape -s -L 0
```

When you do not specify the backup level on the command line, **ontape** prompts you to enter it. [Figure 13-1](#) illustrates a simple **ontape** backup 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 13-1
*Example of a
Simple Backup
Created with
ontape*

Backing Up Raw Tables

You can use **ontape** to back up a raw table. For more information, see [“Restoring Raw Tables” on page 14-16](#).

When the Logical-Log Files Fill During a Backup

When the logical log fills during a backup, the console displays a message and the backup 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 the database 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 a Backup” on page 13-6](#).

When LTAPEDEV and TAPEDEV are identical, assign a different value to the logical-log tape device (LTAPEDEV) and initiate a logical-log-file backup. Otherwise, your options are to either leave normal database server processing suspended until the backup completes or cancel the backup.

When Only One Tape Device Is Available

When you create a backup with the only available tape device, you cannot back up any logical-log files until you complete the backup. When the logical-log files fill during the backup, normal database server processing halts. You can either abort the backup (using **CTRL-C** *only*) to free the tape device and back up the logical logs to continue processing, or leave normal processing suspended until the backup completes.

You can take steps to prevent this situation. The section [“Starting an Automatic Logical-Log Backup” on page 13-16](#) describes these steps.

When a Backup Terminates Prematurely

When you cancel or interrupt a backup, sometimes the backup progresses to the point where you can consider it complete. When listed in the monitoring information, as described in [“Monitoring Backup History Using oncheck” on page 13-13](#), you know the backup completed.

Monitoring Backup History Using oncheck

You can monitor the history of your last full-system backup using **oncheck**.

Execute **oncheck -pr** to display reserved-page information for the root dbospace. The last pair of reserved pages contains the following information for the most recent backup:

- Backup level (0, 1, or 2)
- Effective date and time of the backup
- Time stamp describing when the backup began (expressed as a decimal)
- ID number of the logical log that was current when the backup began
- Physical location in the logical log of the checkpoint record (that was written when the backup began)

The effective date and time of the backup equals the date and time of the checkpoint that this backup took as its starting point. This date and time could differ markedly from the time when the backup process was started.

For example, when no one accessed the database server after Tuesday at 7 P.M., and you create a backup Wednesday morning, the effective date and time for that backup is Tuesday night, the time of the last checkpoint. In other words, when there has been no activity after the last checkpoint, the database server does not perform another checkpoint at the start of the backup.

Backing Up Logical-Log Files with *ontape*

You must only use **ontape** to back up logical-log files when you use **ontape** to make your backup tapes.

In addition to backing up logical-log files, you can use **ontape** to 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 your *Administrator's Guide*.

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
- Whether you want to perform an automatic or continuous backup

For more information on these issues, see [“What Is a Logical-Log Backup?” on page 1-5](#).

Using Blobspace **TEXT** and **BYTE** Data Types and Logical-Log Files

You must keep the following two points in mind when you use **TEXT** and **BYTE** data types in a database that uses transaction logging:

- To ensure timely reuse of blobpages, back up logical-log files. When users delete **TEXT** or **BYTE** values in blobspaces, the blobpages do not become freed for reuse until you free the log file that contains the delete records. To free the log file, you must back it up.
- When you must back up an unavailable blobspace, **ontape** skips it and makes it impossible to recover the **TEXT** or **BYTE** values when it becomes necessary. (However, blobpages from deleted **TEXT** or **BYTE** values do become free when the blobspace becomes available even though the **TEXT** or **BYTE** values were 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 logical-log file that records the event is not the current logical-log file. For information on switching logical-log files, see your *Administrator's Guide*.

Using /dev/null When You Do Not Need to Recover

When you decide that you do not need to recover transactions or administrative database activities between backups, you can set the database 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 database server manages up to the point of your most recent backup 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 14-3.](#)) You cannot perform partial restores or restore when the database server is online.

When you set LTAPEDEV to **/dev/null**, the database server marks a logical-log file as backed up (status B) as soon as it becomes full. The database server can then reuse that logical-log file without waiting for you to back it up. As a result, the database server does not preserve any logical-log records.

Fast recovery and rolling back transactions are not impaired when you use **/dev/null** as your log-file backup device. For a description of fast recovery, see your *Administrator's Guide*. For information about rolling back transactions, see the ROLLBACK WORK statement in the *IBM Informix Guide to SQL: Syntax*.

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 logical-log file because it has a *used* status. For more information on monitoring the status of logical-log files, see your *Administrator's Guide*.

Starting an Automatic Logical-Log Backup

The database server can operate online when you back up logical-log files. To back up all full logical-log files, use the **-a** option of the **ontape** command.

Requesting a Logical-Log Backup

—▶ — -a —▶

The **-a** option backs up all full logical-log files and prompts you with an option to switch the logical-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, the database server finishes the backup and then returns control to you. Any other full logical-log files receive a used status.

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 logical-log files and start backups when the logical-log files become full, you can start a continuous backup.

When you start a continuous backup, the database server automatically backs up each logical-log file as it becomes full. When you perform continuous logical-log file backups, the database server protects you against ever losing more than a partial logical-log file, even in the worst case media failure when a chunk that contains logical-log files fails.

With continuous backups you also do not need to remember to back up the logical-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.

Starting Continuous Backups



The **-c** option initiates continuous backup of logical-log files. The database server backs up each logical-log file as it becomes full. Continuous backup does not back up the current logical-log file.

The database server can operate in online mode when you start continuous backups. To start continuous logging, execute the following command:

```
ontape -c
```

When the tape mounted on `LTAPEDEV` becomes full before the end of the logical-log file, the database server prompts the operator for a new tape.

Ending a Continuous Logical-Log Backup

To end continuous logical-log backup, press the Interrupt key (CTRL-C).

When you press the Interrupt key while the database 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 by the database server.

When you press the Interrupt key while the database server waits for a logical-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 are marked as backed up by the database server.

When you press the Interrupt key while the database server performs a continuous backup to a remote device, any logical-log files that were backed up during this operation can or cannot reside on the tape, and are not marked as backed up by the database server (a good reason why you should 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. However, consider the following issues when you choose a logical-log backup device:

- When the logical-log device differs from the backup device, you can plan your backups without considering the competing needs of the backup 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 data up to the point of your most recent backup tape. You cannot restore work done after the backup. See the warning about setting LTAPEDEV to **/dev/null** in [“Using /dev/null When You Do Not Need to Recover” on page 13-15](#).
- 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.

Restoring with ontape

In This Chapter	14-3
Choosing the Type of Physical Restore	14-3
A Full-System Restore	14-3
Restoring Selected Dbspaces, Blobspaces, and Sbspaces	14-4
Choosing a Cold, Warm, or Mixed Restore.	14-4
A Cold Restore	14-4
A Warm Restore	14-5
A Mixed Restore	14-5
Performing a Restore	14-6
Restoring the Whole System.	14-8
Gathering the Appropriate Tapes	14-8
Backup Tapes	14-8
Logical-Log Tapes	14-8
Deciding on a Complete Cold or a Mixed Restore	14-9
Verifying Your Database Server Configuration	14-9
Setting Shared-Memory Parameters to Maximum Assigned Value.	14-9
Setting Mirroring Configuration to Level-0 Backup State	14-10
Ensuring That Needed Devices Are Available	14-10
Performing a Cold Restore	14-10
Salvaging Logical-Log Files	14-11
Mounting Tapes During the Restore	14-11
Restoring Logical-Log Files	14-11
Bringing the Database Server Online When the Restore Is Over.	14-13

Restoring Selected Storage Spaces	14-14
Gathering the Appropriate Tapes	14-14
Backup Tapes	14-14
Logical-Log Tapes	14-14
Ensuring That Needed Device Are Available.	14-15
Backing Up Logical-Log Files	14-15
Performing a Warm Restore	14-15
Restoring Raw Tables	14-16
Renaming Chunks During a Restore	14-16
Key Considerations	14-16
New Chunk Requirements	14-17
Examples of Renaming Chunks During a Restore	14-18
Renaming Chunks with Command Line Options	14-18
Renaming Chunks with a File	14-18
Renaming Chunks While Specifying Other Options	14-19
Renaming a Chunk to a Nonexistent Device	14-19

In This Chapter

This section provides instructions for restoring data using **ontape** for the following procedures:

- A full-system restore
- A restore of selected dbspaces, blobspaces, and sbspaces

Before you start restoring data, you must understand the concepts in [“What Is a Restore?” on page 1-8](#). As explained in that section, a complete recovery of database server data generally consists of a physical restore and a logical restore.

Choosing the Type of Physical Restore

If a failure causes the database server to go offline, you must restore all the database server data. This type of restore is a *full-system* restore. When the failure did not cause the database server to go offline, you can restore only the storage spaces that failed. For illustrations of the restore types, see [“What Are Warm, Cold, and Mixed Restores?” on page 1-8](#).

A Full-System Restore

When your database server goes offline 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 that contains the physical log
- A dbspace that contains 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 database server manages. You must start a full-system restore with a *cold restore*. See [“Choosing a Cold, Warm, or Mixed Restore” on page 14-4](#).

Restoring Selected Dbspaces, Blobspaces, and Sbspaces

When your database server *does not* go offline because of a disk failure or corrupted data, the damage occurred to a noncritical dbspace, blobspace, or sbspace.

When you do not need to restore a critical dbspace, you can restore only those storage spaces that contain a damaged chunk or chunks. When a media failure occurs in one chunk of a storage space that spans multiple chunks, all active transactions for that storage space must terminate before the database server can restore it. You can start a restore operation before the database server finishes the transactions, but the restore becomes delayed until the database server verifies that you finished all transactions that were active at the time of the failure.

Choosing a Cold, Warm, or Mixed Restore

When you restore the database server data, you must decide whether you can do it while the database server is offline or online. This decision depends in part on the data that you intend to restore.

A Cold Restore

Perform a *cold restore* while the database server is offline. It consists of both a physical restore and a logical restore. You must perform a cold restore to restore any critical dbspaces.

The database server is offline 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.

You can rename chunks by specifying new chunks paths and offsets during a cold restore. This option is useful if you need to restore storage spaces to a different disk from the one on which the backup was made. You can rename any type of chunk, including critical chunks and mirror chunks. For more information, see [“Renaming Chunks During a Restore” on page 14-16](#).

A Warm Restore

A *warm restore* restores noncritical storage spaces while the database server is in online or quiescent mode. It consists of one or more physical restore operations (when you restore multiple storage spaces concurrently), a logical-log backup, and a logical restore.

During a warm restore, the database server replays backed-up logical-log files for the storage spaces that you restore. To avoid overwriting the current logical log, the database 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 or the number of log files being replayed, whichever is smaller. For information on how the database server looks for temporary space, see the discussion of `DBSPACETEMP` in your *Administrator's Guide*.



Warning: Make sure enough temporary space exists for the logical-log portion of the warm restore; the maximum amount of temporary space that the database server needs equals the size of all the logical-log files.

A Mixed Restore

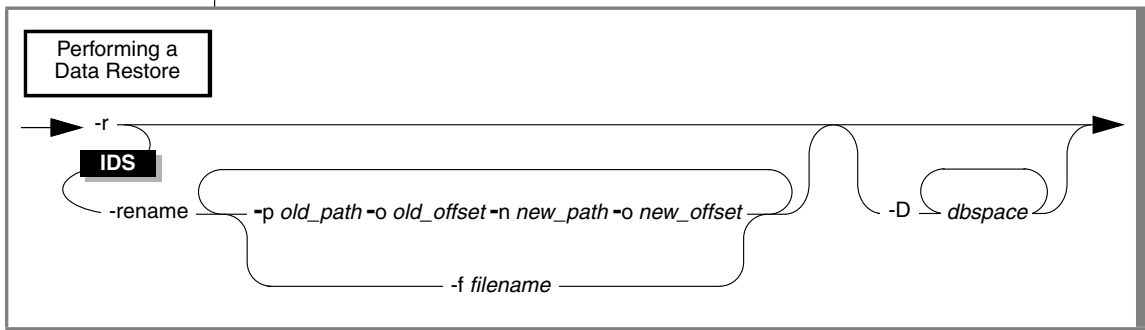
A *mixed restore* is a cold restore followed by a warm restore. A mixed restore restores some storage spaces during a cold restore (the database server is offline) and some storage spaces during a warm restore (the database server is online). You could do a mixed restore when you perform a full-system restore, but you need to provide access to a particular table or set of tables as soon as possible. In this case, perform a cold restore to restore the critical dbspaces and the dbspaces that contain the important tables.

A cold restore takes less total time to restore all your data than a mixed restore, even though the database server is online 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 the database server to go *offline* for less time than a cold restore.

The dbspaces not restored during the cold restore do not become available until after the database server restores them during a warm restore, even though a critical dbspace possibly did not damage them.

Performing a Restore

Use the **-r** option to perform a full physical and logical restore of the database server data with **ontape**. Use the **-D** option to restore selected storage spaces. Use the **-rename** option to rename chunks during the restore.



-r directs **ontape** to perform a data restore (both physical and logical). The **-r** option restores data from the backup tape and the logical-log backup tapes you created after (and including) your last level-0 backup.

-D directs **ontape** to restore only the storage spaces you specify. The database server must go into online or quiescent mode to do a warm restore. When you use the **-D** option, you can restore selected storage spaces.

dbspace is the name of a storage space to restore.

- rename** directs **ontape** to rename the specified chunks.
- p *old_path*** specifies the chunk to be renamed and its new location. Use to
-o *old_offset* rename one or more chunks at one time.
-n *new_path*
-o *new_offset* The variables for this element are:
- *old_path* is the current path and filename of the chunk
 - *old_offset* is the current offset of the chunk, in kilobytes
 - *new_path* is the new path and filename of the chunk
 - *new_offset* is the new offset of the chunk
- f *filename*** specifies a file containing the names and offsets of chunks to be renamed and their new locations. Use to rename a large number of chunks at one time.
- The filename can be any valid UNIX or Windows filename, including simple (**listfile_1**), relative (**../backup_lists/listfile_2** or **..\backup_lists\listfile2**), and absolute (**/usr/informix/backup_lists/listfile3** or **c:\informix\backup_lists\listfile3**) filenames.
- In the file, list the old chunk pathname and offset and the new chunk pathname and offset, with a blank space or a tab between each item. Put information for each chunk on a separate line. Blank lines are ignored. Begin comment lines with a # symbol.

When you do not specify the **-D** option, **ontape** performs a full-system restore. The database server must go offline to do a full-system restore. For more information, see [“Restoring Selected Storage Spaces” on page 14-14](#).

For more information on renaming chunks during a restore, see [“Renaming Chunks During a Restore” on page 14-16](#).

Restoring the Whole System

This section outlines the steps you need to perform to restore your entire 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 database server configuration.
4. Perform a cold restore.

Familiarize yourself with these instructions before you attempt a full-system restore.

Gathering the Appropriate Tapes

Gather the appropriate backup and logical-log tapes.

Backup Tapes

Before you start your restore, gather together all the tapes from your latest level-0 backup that contain the storage spaces you are restoring and any subsequent level-1 or level-2 backups.

Identify the tape that has the latest level-0 backup 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 after the latest level-0 backup of the storage spaces you are restoring.

Deciding on a Complete Cold or a Mixed Restore

As mentioned in [“Choosing a Cold, Warm, or Mixed Restore” on page 14-4](#), when you restore your entire database server, you can restore the critical dbspaces (and any other storage spaces you want to come online quickly) during a cold restore, and then restore the remaining storage spaces during a warm restore. Decide before you start the restore if you want a completely cold, or mixed, restore.

Verifying Your Database 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 database server configuration must remain compatible with, and accommodate, all parameter values assigned after the time of the most recent backup.

For guidance, use the copies of the configuration file that you create at the time of each backup. However, do not set all current parameters to the same values as were recorded at the last backup. Pay attention to the following three groups of parameters:

- Shared-memory parameters
- Mirroring parameters
- Device parameters

Setting 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 backup. For example, when you decreased the value of USERTHREADS from 45 to 30 sometime after the level-0 backup, you must begin the restore with USERTHREADS set at 45, and not at 30, even though the configuration file copy for the last backup 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 backup, 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.)

Setting Mirroring Configuration to Level-0 Backup State

Verify that your current mirroring configuration matches the configuration that was in effect at the time of the last level-0 backup. Because it is recommended that you create a level-0 backup after each change in your mirroring configuration, this creates no problems. The most critical parameters are the mirroring parameters that appear in the ONCONFIG configuration file, MIRRORPATH and MIRROROFFSET.

Ensuring That Needed Devices Are Available

Verify that the raw devices or files that you used for storage (of the storage spaces being restored) after the level-0 backup are available.

For example, when you dropped a dbspace or mirroring for a dbspace after your level-0 backup, you must make the dbspace or mirror chunk device available to the database server when you begin the restore. When the database 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 backup, you must make the chunk device available to the database server when it begins to roll forward the logical logs.

Performing a Cold Restore

To perform a cold restore, the database server must be offline.

You must log in as user **informix** or **root** to use **ontape**. Execute the following **ontape** command to restore all the storage spaces:

```
ontape -r
```

When you perform a mixed restore, you restore only some of the storage spaces during the cold restore. You must restore at least all the critical dbspaces, as the following example shows:

```
ontape -r -D rootdbs llogdbs plogdbs
```


Salvaging Logical-Log Files

Before the restore starts, the console prompts you to salvage the logical-log files on disk. To salvage the logical-log files, use a new tape. It saves log records that you did not back up and enables you to recover your database server data up to the point of the failure.

The following example shows a log salvage:

```
...
Continue restore? (y/n) y
Do you want to back up the logs? (y/n) y

Please mount tape 1 on /dev/ltapedev and press Return to continue.
Would you like to back up any of logs 31 - 32? (y/n) y
Logical logs 31 - 32 may be backed up.
Enter the id of the oldest log that you would like to backup? 31

Please label this tape as number 1 in the log tape sequence.

This tape contains the following logical logs:
    31-32
Log salvage is complete, continuing restore of archive.
Restore a level 1 archive (y/N) y
Ready for level 1 tape
...
```

Mounting Tapes During the Restore

During the restore, **ontape** prompts you to mount tapes with the appropriate backup files.

Restoring Logical-Log Files

When you perform a mixed restore, you must restore all the logical-log files backed up after the last level-0 backup.

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 data only up to the state it was in at the time of your last backup. For more information, see [“Backing Up Logical-Log Files with ontape” on page 13-14.](#)

The following example shows a full restore:

```
ontape -r

Please mount tape 1 on /dev/tapedev and press Return to continue.

Archive Tape Information

Tape type:          Archive Backup Tape
Online version:     IBM Informix Dynamic Server, Version 9.40.UC1
Archive date:       Mon Jun 03 08:32:22 2003
User id:            informix
Terminal id:        /dev/pts/14
Archive level:      0
Tape device:        /dev/tapedev
Tape blocksize (in k): 50
Tape size (in k):   10000
Tape number in series: 1

Spaces to restore: 1 [rootdbs]

Archive Information
...
Initialization Time      05/15/2003 15:41:47
System Page Size         2048
Version                  12
Archive CheckPoint Time  06/03/2000 08:32:25

Dbspaces
number  flags  fchunk  nchunk  flags  owner
name
1       1      1       1       N      informix
rootdbs

Chunks
chk/dbs  offset  size  free  bpages  flags  pathname
1  1      50      25000 13512      PO-
/dev/raws/rootdbs

Continue restore? (y/n) y
Do you want to back up the logs? (y/n) n

Restore a level 1 archive (y/n) y
Ready for level 1 tape

Please mount tape 1 on /dev/tapedev and press Return to continue.
...
Archive Level: 1
Tape device:    /dev/tapedev
Tape blocksize (in k): 50
Tape size (in k): 10000
Tape number in series: 1

Restore a level 2 archive (y/n) y
```

```
Ready for level 2 tape

Please mount tape 1 on /dev/tapedev and press Return to continue.
...
Archive Level: 2
Tape device:    /dev/tapedev
Tape blocksize (in k): 50
Tape size (in k): 10000
Tape number in series: 1

Do you want to restore log tapes? (y/n) y

Roll forward should start with log number 31
Please mount tape 1 on /dev/ltapedev and press Return to continue.
Do you want to restore another log tape? (y/n) y

Please mount tape 2 on /dev/ltapedev and press Return to continue.
Do you want to restore another log tape? 9y/n) n

Program over.
```

Bringing the Database Server Online When the Restore Is Over

At the end of the cold restore, the database server is in quiescent mode. You can bring the database server online at this point and continue processing as usual.

When you restored only some of your storage spaces during the cold restore, you can start a warm restore of the remaining storage spaces after you bring the database server online.

Restoring Selected Storage Spaces

This section outlines the steps that you must perform during a restore of selected storage spaces with **ontape** while the database server is in online or quiescent mode (a warm restore). During a warm restore, you do not need to worry about shared-memory parameters as you do for cold restores.

The following list describes the main steps in a warm restore:

1. Gather the appropriate tapes.
2. Verify your database server configuration.
3. Back up logical-log files.
4. Perform a warm restore.

Before you attempt a restore, familiarize yourself with these instructions.

Gathering the Appropriate Tapes

Gather the appropriate backup and logical-log tapes.

Backup Tapes

Before you start your restore, gather together all the tapes from your latest level-0 backup that contain the storage spaces you are restoring and any subsequent level-1 or level-2 backups.

Logical-Log Tapes

Gather together all the logical-log tapes from the logical-log backup after the latest level-0 backup of the storage spaces you are restoring.

Ensuring That Needed Device Are Available

Verify that storage devices and files are available before you begin a restore. For example, when you drop a dbspace or mirroring for a dbspace after your level-0 backup, you must ensure that the dbspace or mirror chunk device is available to the database server when you begin the restore. If the storage device is not available, the database server cannot write to the chunk and the restore fails.

When you add a chunk after your last backup, you must ensure that the chunk device is available to the database server when it rolls forward the logical logs.

Backing 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 with ontape” on page 13-14](#).

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.
```

Performing a Warm Restore

To perform a warm restore, the database server must operate in online or quiescent mode.

You must log in as user **informix** or **root** to use **ontape**. To restore selected storage spaces, execute the **ontape** command, with the options that the following example shows:

```
ontape -r -D dbspace1 dbspace2
```

You cannot restore critical dbspaces during a warm restore; you must restore them as part of a cold restore, described in [“Restoring the Whole System” on page 14-8](#).

During the restore, **ontape** prompts you to mount tapes with the appropriate backup files.

At the end of the warm restore, the storage spaces that were down go online.

Restoring Raw Tables

When you use **ontape** to restore a raw table, it contains only data that existed on disk at the time of the backup. Because raw tables are not logged, any changes that occurred since the last backup cannot be restored. For more information, see [“Backing Up Raw Tables” on page 13-12](#) and the *Administrator’s Guide*.

IDS

Renaming Chunks During a Restore

You can rename chunks during a cold restore with **ontape**. This option is useful if you need to restore storage spaces to a different disk from the one on which the backup was made. You can rename any type of chunk, including critical chunks and mirror chunks.

For the syntax of renaming chunks with **ontape**, see [“Performing a Restore” on page 14-6](#).

Tip: *If you use symbolic links to chunk names, you might not need to rename chunks; you need only edit the symbolic name definitions. For more information, see the “Administrator’s Guide.”*



Key Considerations

During a cold restore, **ontape** performs the following validations to rename chunks:

1. It validates that the old chunk pathnames and offsets exist in the archive reserved pages.
2. It validates that the new chunk pathnames and offsets do not overlap each other or existing chunks.

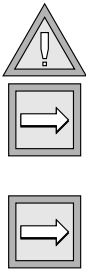
3. If renaming the primary root or mirror root chunk, it updates the ONCONFIG file parameters ROOTPATH and ROOTOFFSET, or MIRRORPATH, and MIRROROFFSET. The old version of the ONCONFIG file is saved as **\$ONCONFIG.localtime**.
4. It restores the data from the old chunks to the new chunks (if the new chunks exist).
5. It writes the rename information for each chunk to the online log.

If either of the validation steps fails, the renaming process stops and **ontape** writes an error message to the **ontape** activity log.

Warning: Perform a level-0 archive after you rename chunks; otherwise your next restore will fail.

Important: If you add a chunk after performing a level-0 archive, that chunk cannot be renamed during a restore. Also, you cannot safely specify that chunk as a new path in the mapping list.

Important: Renaming chunks for database servers participating in HDR involves a significant amount of time offline for both database servers. For more information, see the “Administrator’s Guide.”



New Chunk Requirements

To rename a chunk, follow these guidelines for new chunks:

- The new chunk does not need to exist
You can install the new chunk later and perform a warm restore of a storage space containing it. If you specify a nonexistent chunk, **ontape** records the rename information in the chunk reserved pages, but does not restore the data. The renamed (but not restored) chunks have a status of offline, designated by **N**, in the **onstat -d** chunk status command output.
- New chunks must have the proper permissions.
Rename operations fail unless the chunks have the proper permissions. For more information, see the *Administrator’s Guide*.

Examples of Renaming Chunks During a Restore

To rename a chunk, provide the old chunk location and the new chunk location, either at the command line or in a file.

The following table lists example values for two chunks that are used in the examples in this section.

Element	Value for First Chunk	Value for Second Chunk
old path	/chunk1	/chunk2
old offset	0	10000
new path	/chunk1N	/chunk2N
new offset	20000	0

Renaming Chunks with Command Line Options

To rename the chunks by supplying information on the command line, use this command:

```
ontape -r -rename -p /chunk1 -o 0 -n /chunk1N -o 20000
        -rename -p /chunk2 -o 10000 -n /chunk2N -o 0
```

Perform a level-0 archive after the rename and restore operation is complete.

Renaming Chunks with a File

To rename the chunks by supplying a file named **listfile**, use this command:

```
ontape -r -rename -f listfile
```

The contents of the **listfile** file are:

```
/chunk1 0 /chunk1N 20000
/chunk2 10000 /chunk2N 0
```

Perform a level-0 archive after the rename and restore operation is complete.

Renaming Chunks While Specifying Other Options

To rename the chunks using command-line options while performing a restore of **dbspace1** and **dbspace2**, use the following command:

```
ontape -r -rename -p /chunk1 -o 0 -n /chunk1N -o 20000
        -rename -p /chunk2 -o 10000 -n /chunk2N -o 0
        -D dbspace1 dbspace2
```

Alternatively, to rename the chunks using file while performing a restore of **dbspace1** and **dbspace2**, use the following command:

```
ontape -r -rename -f listfile -D dbspace1 dbspace2
```

Perform a level-0 archive after the rename and restore operation is complete.

Renaming a Chunk to a Nonexistent Device

To rename a chunk to a device that does not yet exist, you specify the new path name, but you do not restore its storage spaces until after you install the physical device. This option is useful if you need to rename a chunk and it is convenient to perform a cold restore before you install the new device. When the new chunk device is ready, you can perform a warm restore of a storage space onto it.

You can combine renaming chunks with existing devices and renaming chunks with nonexistent devices in the same rename operation. This example shows how to rename a single chunk to a nonexistent device name.

The following table lists example values for the chunks used in this example.

Storage Space	Old Chunk Path	Old Offset	New Chunk Path	New Offset
sbspace1	/chunk3	0	/chunk3N	0

To rename a chunk to a nonexistent device

1. Rename the chunk:

```
ontape -r -rename -p /chunk3 -o 0 -n /chunk3N -o 0
```

2. When the following prompt appears, enter **y** to continue: The chunk /chunk3N does not exist. If you continue, the restore may fail later for the dbspace which contains this chunk. Continue without creating this chunk? (y/n)

The chunk **/chunk3** is renamed to **/chunk3N**, but the data has not yet been restored to **/chunk3N**.

3. Perform a level-0 archive.
4. Add the physical device for **/chunk3N**.
5. Perform a warm restore of **sbspace1**:

```
ontape -r -D sbspace1
```

6. Perform a level-0 archive.

Troubleshooting

This appendix lists some error messages you can receive during a backup or restore, describes under what circumstances the errors might occur, and possible solutions or work-arounds.

Corrupt Page During an Archive

The message Archive detects that page is corrupt indicates that page validation failed.

During an archive, the database server validates every page before writing it to the archive device. This validation checks that the elements on the page are consistent with the expected values. When a page fails this validation, a message similar to the following is written to the **online.log** file:

```
15:06:37 Assert Failed: Archive detects that page
0xc00021 is corrupt.
15:06:37 IBM Informix Dynamic Server Version 9.40.UC1
15:06:37 Who: Session(25, informix@cronus, 67612,
1085259772)
          Thread(50, arcbackup1, 40acf758, 4)
          File: rsarcbu.c Line: 2549
15:06:37 stack trace for pid 67367 written to
/tmp/af.41ad7b9
15:06:37 See Also: /tmp/af.41ad7b9
```

The page number is printed in hexadecimal. The format for page number is 0xCCCCPPPPP where CCC represents the chunk number, and PPPPP represents the page number. For this example, the corrupted page is in chunk 0xc (12 decimal) and page 0x21 (33 decimal). The archive aborts after detecting 10 corrupt pages. The **online.log** file displays the full error message, including the page address, for the first 10 errors. Subsequently, only the count of the number of corrupt pages is put in to the **online.log**.

When you receive this message, identify which table the corrupt page belongs to by examining the output of the **oncheck -pe** command. To determine the extent of the corruption, execute the **oncheck -cID** command for that table.

A corrupt page is saved onto the backup media. During a restore, the corrupt page are returned in its corrupt form. There are no errors messages to the **online.log** when corrupt pages are restored, just when they are archived.

Log Backup Already Running

When using ON-Bar to create a backup, the message log backup is already running in the **bar_act.log** file, or the message Process exited with return code 152 in the **online.log** file do not indicate a problem. They can appear under the following circumstances:

- The ALARMPROGRAM configuration parameter is set to **log_full.sh**. Periodically, events cause **log_full.sh** to trigger the **onbar -b -l** command. If a log fills while the **onbar -b -l** command is running, then ON-Bar backs up that log as well. If the backup has not completed by the time of the next event trigger, it generates a warning in the **bar_act.log** file. At the time of the next event trigger, the log backup can continue.
- A level-0 archive (especially when started with the **-w** option) first archives the database and then automatically start the **onbar -b -l** command to back up any logical logs that are currently full and not yet backed up. There might not be a **log_full.sh** message in **online.log**, because the **onbar -b -l** command is started directly.

- When you mount a new tape after filling a previous tape, a **log_full.sh** event is scheduled but not triggered. As soon as the next log fills and generates an event trigger in the **log_full.sh** file, all available logs are archived. You can force the archive by running **onbar -l** or force **log_full.sh** to be triggered by running **onmode -l**.

No Server Connection During a Restore

During a whole system restore with ON-Bar, the error archive api error: no server connection. might appear in the **bar_act.log** file. ON-Bar then connects to the storage manager successfully, but eventually fails with the error archive api error: not yet open.

The **bar_act.log** file contains information similar to the following messages:

```
2000-03-09 10:51:06 19304 19303 /usr/informix/bin/onbar_d -r -w
2000-03-09 10:51:09 19304 19303 ERROR: Unable to start the
physical restore:
Archive API error: no server connection.
2000-03-09 10:51:09 19304 19303 Successfully connected to Storage
Manager.
2000-03-09 10:51:36 19304 19303 Process 19304 received signal 3.
Process will
exit after cleanup.
2000-03-09 10:59:13 19811 19810 /usr/informix/bin/onbar_d -r -w
2000-03-09 10:59:16 19811 19810 ERROR: Unable to start the
physical restore:
Archive API error: no server connection.
2000-03-09 10:59:16 19811 19810 Successfully connected to Storage
Manager.
2000-03-09 11:01:12 19811 19810 Begin cold level 0 restore llog1.
2000-03-09 11:01:12 19811 19810 ERROR: Unable to write restore
data to the
database server: Archive API error: not yet open.
```

To solve this problem, check if the database server is still running. If it is, shut down the database server and run the command again.

Dropping a Database Before a Restore

If you perform a level-0 archive using ON-Bar and a storage manager, then drop a database, and then perform a restore with the **onbar -r** command, the database remains dropped. The restore salvages the logs and the logs contains the DROP DATABASE statement. When the logs are salvaged, or replayed, the database is dropped.

To prevent this situation, perform a physical restore using the **onbar -r -p** command, and then a logical restore using the **onbar -r -l** command. This sequence does not salvage the logs and does restore the database.

No dbspaces or blobspaces During a Backup or Restore

If the emergency boot file, **ixbar.servernum**, does not have the correct entries for objects in the backup, the message *There are no DB/BLOBspaces to backup/restore* appears in **bar_act.log** file during a restore started with the **onbar -r** or **onbar -r -w** command.

This error can appear under the following circumstances:

- During an external restore, if the emergency boot file was not copied from the source system.
- If the emergency boot file was recreated after the archive backup was made. The previous file is saved in the form: **ixbar.xx.xxxx**.
- An attempt to execute the **onbar -r -w** command with a backup that is not a full system backup.

onstat Command Reference

This appendix describes forms of the **onstat** command that are relevant to ON-Bar and **ontape**. The **onstat** utility reads shared-memory structures and provides statistics about the database server that are accurate at the instant that the command executes. For general information about **onstat**, refer to the *Administrator's Reference*.

onstat -d

Use the **-d** option to display information for chunks in each storage space. You can interpret output from this option as follows. The first section of the display describes the storage spaces:

<i>address</i>	Is the address of the storage space in the shared-memory space table	
<i>number</i>	Is the unique ID number of the storage space assigned at creation	
<i>flags</i>	Uses the following hexadecimal values to describe each storage space:	
	0x00000000	Mirror not allowed and dbspace is unmirrored
	0x00000001	Mirror is allowed and dbspace is unmirrored
	0x00000002	Mirror is allowed and dbspace is mirrored
	0x00000004	Down
	0x00000008	Newly mirrored
	0x00000010	Blobspace
	0x00000020	Blobspace on removable media
	0x00000040	Blobspace is on optical media
	0x00000080	Blobspace is dropped
	0x00000100	Blobspace is the optical STAGEBLOB
	0x00000200	Space is being recovered
	0x00000400	Space is fully recovered
	0x00000800	Logical log is being recovered

	0x00001000	Table in dbspace is dropped
	0x00002000	Temporary dbspace
	0x00004000	Blobspace is being backed up
	0x00008000	Sbspace
	0x0000a001	Temporary sbspace
	0x00010000	Physical or logical log changed
	0x00020000	Dbspace or chunk tables have changed
<i>fchunk</i>	Is the ID number of the first chunk	
<i>nchunks</i>	Is the number of chunks in the storage space	
<i>flags</i>	Uses the following letter codes to describe each storage space:	
	Position 1:	M Mirrored N Not Mirrored
	Position 2:	R Being recovered P Physically recovered, waiting for logical recovery L Being logically recovered X Newly mirrored D Disabled
	Position 3:	B Blobspace S Sbspace T Temporary dbspace
	Position 4:	B Has big chunks (IDS)
<i>owner</i>	Is the owner of the storage space	
<i>name</i>	Is the name of the storage space	

The line immediately following the storage-space list includes the number of active spaces (the current number of dbspaces in the database server instance including the rootdbs) and the number of total spaces.

Active spaces refers to the current number of storage spaces in the database server instance including the rootdbs. **Total** refers to total *allowable* spaces for this database server instance.

The second section of the **onstat -d** output describes the chunks:

<i>address</i>	Is the address of the chunk		
<i>chk/dbs</i>	Is the chunk number and the associated space number		
<i>offset</i>	Is the offset into the file or raw device in pages		
<i>size</i>	Is the size of the chunk in page		
<i>free</i>	Is the number of free pages in the chunk for a dbspace		
	For a blobspace, a tilde indicates an approximate number of free blobpages.		
	For an sbpace, indicates the number of free pages of user data space and total user data space.		
<i>bpages</i>	Is the size of the chunk in blobpages		
	Blobpages can be larger than disk pages; therefore, the bpages value can be less than the size value.		
	For an sbpace, is the size of the chunk in sbpages		
<i>flags</i>	Provides the chunk status information as follows:		
	Position 1:	P	Primary
		M	Mirror
	Position 2:	O	Online
		D	Down
		X	Newly mirrored
		I	Inconsistent

	R	Being recovered
	N	Renamed and either down or inconsistent (IDS)
Position 3:	-	Dbospace
	B	Blobspace
	S	Sbspace
Position 4:	B	Big chunk (IDS)

pathname Is the pathname of the physical device

The line immediately following the chunk list displays the number of active chunks (including the root chunk) and the total number of chunks.

For information on how to solve problems indicated by flags in position 2 of both the storage space and chunk sections, see [“Monitoring Restores” on page 6-8](#).

XPS

onstat -g bus

Prints current backup scheduler sessions, backups in progress, and backups to be performed. Issue from any coserver.

XPS

onstat -g bus_sm

Prints current storage manager configuration and active work. Issue from any coserver.

onstat -l

Use the **-l** option to display information about physical and logical logs. You can interpret output from this option as follows. The first section of the display describes the physical-log configuration:

<i>buffer</i>	Is the number of the physical-log buffer
<i>bufused</i>	Is the number of pages of the physical-log buffer that are used
<i>bufsize</i>	Is the size of each physical-log buffer in pages
<i>numpages</i>	Is the number of pages written to the physical log
<i>numwrits</i>	Is the number of writes to disk
<i>pages/io</i>	Is calculated as $\text{numpages}/\text{numwrits}$ This value indicates how effectively physical-log writes are being buffered.
<i>phybegin</i>	Is the physical page number of the beginning of the log
<i>physize</i>	Is the size of the physical log in pages
<i>phypos</i>	Is the current position in the log where the next log-record write is to occur
<i>phyused</i>	Is the number of pages used in the log
<i>%used</i>	Is the percent of pages used

The second section of the **onstat -l** display describes the logical-log configuration:

<i>buffer</i>	Is the number of the logical-log buffer
<i>bufused</i>	Is the number of pages used in the logical-log buffer
<i>bufsize</i>	Is the size of each logical-log buffer in pages
<i>numrecs</i>	Is the number of records written
<i>numpages</i>	Is the number of pages written

<i>numwrits</i>	Is the number of writes to the logical log
<i>recs/pages</i>	Is calculated as <code>numrecs/numpages</code> You cannot affect this value. Different types of operations generate different types (and sizes) of records.
<i>pages/io</i>	is calculated as <code>numpages/numwrits</code> You can affect this value by changing the size of the logical-log buffer (specified as <code>LOGBUFF</code> in the <code>ONCONFIG</code> file) or by changing the logging mode of the database (from buffered to unbuffered, or vice versa).

The following fields are repeated for each logical-log file:

<i>address</i>	Is the address of the log-file descriptor																
<i>number</i>	Is logid number for the logical-log file The logid numbers might be out of sequence because either the database server or administrator can insert a log file in-line.																
<i>flags</i>	Provides the status of each log as follows: <table> <tr> <td>A</td><td>Newly added (and ready to use)</td></tr> <tr> <td>B</td><td>Backed up</td></tr> <tr> <td>C</td><td>Current logical-log file</td></tr> <tr> <td>D</td><td>Marked for deletion</td></tr> <tr> <td></td><td>To drop the log file and free its space for reuse, you must perform a level-0 backup of all storage spaces</td></tr> <tr> <td>F</td><td>Free, available for use</td></tr> <tr> <td>L</td><td>The most recent checkpoint record</td></tr> <tr> <td>U</td><td>Used</td></tr> </table>	A	Newly added (and ready to use)	B	Backed up	C	Current logical-log file	D	Marked for deletion		To drop the log file and free its space for reuse, you must perform a level-0 backup of all storage spaces	F	Free, available for use	L	The most recent checkpoint record	U	Used
A	Newly added (and ready to use)																
B	Backed up																
C	Current logical-log file																
D	Marked for deletion																
	To drop the log file and free its space for reuse, you must perform a level-0 backup of all storage spaces																
F	Free, available for use																
L	The most recent checkpoint record																
U	Used																
<i>uniqid</i>	Is the unique ID number of the log																
<i>begin</i>	Is the beginning page of the log file																

<i>size</i>	Is the size of the log in pages
<i>used</i>	Is the number of pages used
<i>%used</i>	Is the percent of pages used
<i>active</i>	Is the number of active logical logs
<i>total</i>	Is the total number of logical logs

The database server uses *temporary logical logs* during a warm restore because the permanent logs are not available then. The following fields are repeated for each temporary logical-log file:

<i>address</i>	Is the address of the log-file descriptor
<i>number</i>	Is logid number for the logical-log file
<i>flags</i>	Provides the status of each log as follows: B Backed up C Current logical-log file F Free, available for use U Used
<i>uniquid</i>	Is the unique ID number of the log
<i>begin</i>	Is the beginning page of the log file
<i>size</i>	Is the size of the log in pages
<i>used</i>	Is the number of pages used
<i>%used</i>	Is the percent of pages used
<i>active</i>	Is the number of active temporary logical logs

Migration

Using Data-Migration Tools for Recovery

If ON-Bar and **ontape** are not working, you can use data-migration utilities, such as **onunload**, the High-Performance Loader (HPL), **onpladm**, or **dbexport**, as a substitute for a backup. For more information on these utilities, see the *IBM Informix Migration Guide*.



Important: *None of the data-migration utilities are coordinated with the information stored in the logical-log files and, unlike backups, they do not save a copy of system-overhead information important to the database server.*



Preparing for a Database Server or Storage-Manager Upgrade

Important: The database server conversion software automatically re-creates the **sysutils** database when you upgrade to the latest version of the database server. All backup and restore information from the old database server version is lost. Backups that you make under the older version of the database server are not compatible with the newer version of the database server.

To prepare for an upgrade

1. Use ON-Bar to perform a level-0 backup of all your data before you upgrade your database server, ISM, or change storage-manager vendors.
2. Save these backups so that you can restore the data in case you need to revert to the old database server version.
3. Before you upgrade, back up the administrative files.
4. After you upgrade the database server, back up all storage spaces and logical logs.

For more information on database server migration, see the *IBM Informix Migration Guide*.

Upgrading Your Storage Manager

If you install a new version of a third-party storage manager, install it before you bring up the database server. Update the **sm_versions** file with the new storage-manager definition. If you have continuous logical-log backup set up on the database server, ON-Bar can start backing up the logical logs soon after the database server comes online. Also make sure that the new storage-manager version is able to read media written with your old version.

Make sure that the storage manager can find the backup objects that ON-Bar requests. Use the **onsmsync** utility to expire old backup history in the **sysutils** database and emergency boot files.

In Extended Parallel Server only, ensure that the **BAR_SM** parameters in the **ONCONFIG** file match the new storage-manager definition. ♦

Changing Storage-Manager Vendors

When you switch storage-manager vendors, the transition can be difficult. Ensure that the new data formats are identical, that a reversion utility is provided, or that you do not use new features that change the data formats. Differences usually occur in the following areas:

- The new storage manager might support different storage devices. If you also upgrade a storage device, make sure the old storage device is available until you successfully back up and restore on the new storage device.
- If you change physical connectivity, such as moving a storage device from a local connection to a network server, make sure the storage manager can still move the data across the network.
- If you use software compression or encryption, make sure all versions of the compression or encryption algorithms are available for restores.
- Ensure that the storage manager can send multiple data streams to storage devices. It also might use a different version of XBSA.

You can switch between certain storage managers more easily than others. For details, contact Technical Support or your vendor.

IDS

Migrating from ontape to ON-Bar

You cannot back up data with **ontape** and restore it using ON-Bar, or conversely because the data storage formats and backup capabilities are different. You can use **ontape** with the database server in online or quiescent mode.

To migrate to ON-Bar

1. Use **ontape** to perform a full backup.
For details, see [Chapter 13, “Backing Up with ontape.”](#)
2. Take the backup media offline to prevent possible reuse or erasure.

3. Configure the storage manager to be used with ON-Bar.
For details, see [Chapter 3, “Configuring the Storage Manager and ON-Bar.”](#)
4. Configure your ON-Bar environment:
 - a. Set ONCONFIG parameters.
 - b. Create the **sm_versions** file with the storage-manager definition.
For details, see [Chapter 9, “Setting ON-Bar Configuration Parameters,”](#) and [“Updating the sm_versions File”](#) on page 3-5.
5. Use ON-Bar (**onbar -b** or **onbar -b -w**) to perform a full backup.
6. Verify the backup with **onbar -v**.
For details, see [Chapter 5, “Verifying Backups.”](#)

Migrating Private ON-Bar Scripts

This section describes the procedures for migrating private ON-Bar scripts after you upgrade the database server version.

If your script searches for ON-Bar process IDs, be aware that the relationship between the **onbar-driver** and the **onbar_d** child processes or **onbar-worker** processes is quite different for Dynamic Server and Extended Parallel Server.

If you reuse a private script for Extended Parallel Server on Dynamic Server, remove the following ON-Bar options that Dynamic Server does not support:

- **-q** (session name)
- **-b -p** (physical-only backup)

◆

IDS

XPS

If you reuse a private script for Dynamic Server on Extended Parallel Server, remove the following ON-Bar options that Extended Parallel Server does not support:

- **-w** (whole-system backup)
- **-c** (current log backup)
- **-C** (continuous-log backup)
- **-RESTART** (restartable restore)

◆

GLS Support

Using GLS with ON-Bar

ON-Bar supports Global Language Support (GLS), which allows users to work in their native language. The language that the client application uses is called the *client locale*. The language that the database uses for its server-specific files is called the *server locale*.

ON-Bar must run on the same computer as the database server. However, you can run ON-Bar in any locale for which you have the supporting message and localization files. For example, if the server locale is English and the client locale is French, you can issue ON-Bar commands in French.

The following command performs a level-0 backup of the dbspaces specified in the file, **tombé**:

```
onbar -b -L 0 -f tombé
```

Windows

On Windows, you cannot use multibyte filenames in backup or restore commands because they are not supported. ♦

The **sysutils** database, the emergency boot files, and the storage-manager boot file are created with the **en_us.8859-1** (default English) locale. The ON-Bar catalog tables in the **sysutils** database are in English. Change the client and database locales to **en_us.8859-1** before you attempt to connect to the **sysutils** database with DB-Access or third-party utilities.

Identifiers That Support Non-ASCII Characters

The *IBM Informix GLS User's Guide* describes the SQL identifiers that support non-ASCII characters. Non-ASCII characters include both 8-bit and multibyte characters. You can use non-ASCII characters in the database names and filenames with the ON-Bar, **ondblog**, and **onutil** commands, and for filenames in the ONCONFIG file.

For example, you can specify a non-ASCII filename for the ON-Bar activity log in BAR_ACT_LOG and a non-ASCII pathname for the storage-manager library in BAR_BSA LIB_PATH.

Identifiers That Require 7-Bit ASCII Characters

You must use 7-bit ASCII characters for the following identifiers:

- Storage-space names
- Database server names

Locale of ON-Bar Messages

All ON-Bar messages appear in the activity log in the client locale except the messages that the database server issues. For example, the part of the message that tells you that a database server error occurred appears in the client locale, and the server-generated part appears in the server locale.

Using the GL_DATETIME Environment Variable with ON-Bar

The database server must know how to interpret and convert the end-user formats when they appear in date or time data that the client application sends. You can use the GL_DATE and GL_DATETIME environment variables to specify alternative date and time formats. If you do not set these environment variables, ON-Bar uses the date and time format of the client locale.

If you perform a point-in-time restore, enter the date and time in the format specified in the GL_DATETIME environment variable if it is set.

Point-in-Time Restore Example

For example, the default date and time format for the French locale, `fr_fr.8859-1`, uses the format "%A %.1d %B %iY %H:%M:%S." The ON-Bar command for a point-in-time restore is as follows:

```
onbar -r -t "Lundi 9 Juin 1997 11:20:14"
```

You can set `GL_DATETIME` to a different date and time format that uses the date, month, two-digit year, hours, minutes, and seconds.

```
%.1d %B %iy %H:%M:%S
```

The ON-Bar command for a point-in-time restore is as follows:

```
onbar -r -t "9 Juin 97 11:20:14"
```

Tip: For more information on how to use GLS and the `GL_DATE` and `GL_DATETIME` environment variables, refer to the “IBM Informix GLS User’s Guide.”



Using GLS with ontape

The **ontape** utility supports GLS in the same way as ON-Bar does. You can specify the database name in the national locale.

Notices

IBM may not offer the products, services, or features discussed in this document in all countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. (enter the year or years). All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

AIX; DB2; DB2 Universal Database; Distributed Relational Database Architecture; NUMA-Q; OS/2, OS/390, and OS/400; IBM Informix[®]; C-ISAM[®]; Foundation.2000[™]; IBM Informix[®] 4GL; IBM Informix[®] DataBlade[®] Module; Client SDK[™]; Cloudscape[™]; Cloudsync[™]; IBM Informix[®] Connect; IBM Informix[®] Driver for JDBC; Dynamic Connect[™]; IBM Informix[®] Dynamic Scalable Architecture[™] (DSA); IBM Informix[®] Dynamic Server[™]; IBM Informix[®] Enterprise Gateway Manager (Enterprise Gateway Manager); IBM Informix[®] Extended Parallel Server[™]; i.Financial Services[™]; J/Foundation[™]; MaxConnect[™]; Object Translator[™]; Red Brick Decision Server[™]; IBM Informix[®] SE; IBM Informix[®] SQL; InformiXML[™]; RedBack[®]; SystemBuilder[™]; U2[™]; UniData[®]; UniVerse[®]; wintegrate[®] are trademarks or registered trademarks of International Business Machines Corporation.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Windows, Windows NT, and Excel are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Other company, product, and service names used in this publication may be trademarks or service marks of others.

Index

Numerics

9.3 features, overview Intro-6

A

Activity log, ON-Bar

checking restore status 6-37

described 2-17

diagnosing failed

verification 5-13

error messages 11-5 to 11-8

finding backup time 6-22

message format 11-3

message numbers 11-4

monitoring backups 4-10

overview 2-17

return codes 11-9 to 11-15

skipped dbspace message 6-16

specifying location 3-11, 3-13, 9-8

specifying progress

messages 3-11, 3-14, 9-16

status of archecker 5-5, 5-6, 9-28

verification message 5-11

warning about /dev/null 9-26

AC_CONFIG environment

variable 5-6, 5-13, 9-5 to 9-6

ac_config.std file 5-6, 9-5 to 9-6

AC_MSGPATH parameter 9-4, 9-5, 9-28

ac_msg.log. *See* archecker message log.

AC_STORAGE directory 5-13

AC_STORAGE parameter 9-4, 9-5

AC_VERBOSE parameter 9-4, 9-6

Administrative files

backing up 4-6

copying before cold restore 6-20, 6-26

external backup 7-5, 7-7

Administrative tasks

backing up after changing

schema 4-13, 13-7

using ISA 4-15

ALARMPROGRAM parameter

alarmprogram.sh or

alarmprogram.bat 9-7

description 3-11, 9-7

setting

log_full.sh or log_full.bat 4-24

no_log.sh or no_log.bat 4-24

alarmprogram.sh script 4-24, 9-7

Altering table type 4-14, 6-39

ANSI compliance

level Intro-16

archecker message log 9-5

archecker utility

AC_CONFIG parameters 9-5

blobspaces 5-10

estimating temporary space 5-8

fixing verification

problems 5-13 to 5-14

interpreting verification

messages 5-11

messages 5-11, 9-5

onmsync 5-13, 5-14

point-in-time verification 5-10

preparing to verify backups 5-6

sbspaces 5-10

sbspaces and blobspaces 5-9

syntax diagram 5-7

temporary files 5-8, 9-6

verification failure, causes of 5-12
 verifying expired backups 5-14
 whole-system backup 5-10
 Autochangers 3-21
 Automatic backups. *See* Continuous
 log backup.

B

Backup
 physical, XPS 4-7
 Backup Scheduler
 definition 2-10
 how it works 4-32
 monitoring status 8-17
 SMI tables overview 10-11
 sysbuobject table 10-12
 sysbuobjses table 10-13
 sysbusession table 10-14
 sysbusm table 10-14
 sysbusmdbspace table 10-15
 sysbusmlog table 10-15
 sysbusmworker table 10-16
 sysbuworker table 10-16
 Backup, ON-Bar
 assigning a session name 4-20
 automatic log backups 9-7
 changing database logging 4-14
 checking data consistency 4-10
 complete, example of 4-16
 current log 2-5, 7-8
 detailed description 4-10 to 4-21
 external 2-5, 7-5
 fake backups 2-5, 4-19
 imported restore 6-32 to 6-35
 incremental, example of 4-16
 ISM catalog 4-14
 large chunk files 4-5
 list of storage spaces 4-16
 logical log 1-5, 2-5, 4-21 to 4-27
 monitoring progress 2-17
 -O option 2-5, 4-12, 4-20
 operational tables 4-21
 physical-only 2-5, 4-12, 4-20
 raw tables 4-21
 salvaging logs 1-6, 4-23, 4-28
 sbspaces, example of 4-18
 scratch tables 4-21

sequence
 Dynamic Server 4-30
 Extended Parallel Server 4-32
 standard backup 2-5
 standard tables 4-21
 static tables 4-21
 storage-manager installation 4-7
 syntax 4-11
 table types 4-14, 4-21
 temp tables 4-21
 verification 2-5, 5-3 to 5-14
 whole system 2-5
See also Continuous log backup;
 Storage spaces; Whole-system
 backup.
 Backup, ontape
 before you create 13-7
 creating 13-11
 if interrupted 13-13
 if logical log fills 13-12
 if terminates prematurely 13-13
 labelling the tape 13-10
 levels 13-6
 monitoring 13-13
 one device 12-5
 Backup, overview
 creating a plan 1-17
 levels 1-4
 planning 1-18
 schedule 1-19, 2-13
 bargroup group 3-7
 bar_action table 10-3
 BAR_ACT_LOG parameter 2-17,
 3-11, 3-13, 9-8
 BAR_BOOT_DIR parameter 3-13,
 9-9
 BAR_BSALIB_PATH
 parameter 3-9, 3-11, 3-13, 9-10
 BAR_DBS_COSVR parameter 3-13,
 9-11
 BAR_HISTORY parameter 3-11,
 3-14, 9-12
 BAR_IDLE_TIMEOUT
 parameter 3-14, 4-32, 9-13
 bar_instance table 10-5
 BAR_LOG_COSVR
 parameter 3-14, 9-13
 BAR_MAX_BACKUP
 parameter 3-11, 4-30, 9-14

BAR_NB_XPORT_COUNT
 parameter 3-11, 9-15
 bar_object table 10-7
 BAR_PROGRESS_FREQ
 parameter 3-11, 3-14, 9-16
 BAR_RETRY parameter 3-11, 3-14,
 9-17
 bar_server table 10-8
 BAR_SM parameter 3-14, 9-19
 BAR_SM_NAME parameter 3-14,
 9-19
 bar_version table 10-8
 BAR_WORKER_COSVR
 parameter 3-14, 9-20
 BAR_WORKER_MAX
 parameter 3-15, 3-17, 9-20
 BAR_XFER_BUFSIZE parameter
 description of 3-15, 9-23
 page size 9-24
 BAR_XFER_BUF_SIZE
 parameter 3-12, 9-22
 BAR_XPORT_COUNT parameter
 description of 3-15, 9-24
 page size 9-24
 Bixbar.hostname.servnum file 2-16
 bldutil.process_id file 9-28
 Blobspaces
 availability for backing up 4-19,
 13-15
 backing up 1-4
 backing up offline 4-20
 optical platters 4-6
 temp space for archiver 5-9
 Block size, ontape
 parameter 12-7
 tape device 12-9
 Blocking, database server 7-5, 7-9
 Boldface type Intro-8

C

Child process, ON-Bar 4-30
 Chunks, renaming
 examples using ON-Bar 6-30
 examples using ontape 14-18
 ON-Bar syntax 6-28
 overview with ON-Bar 6-27
 overview with ontape 14-16

Client locale D-1
Cold restore
 defined 1-9
ON-Bar
 example of 6-19
 renaming chunks during 6-27
 sequence 6-48, 6-52
 setting mode 6-4
ontape
 description of 14-4
 mixed restore 14-5
 performing 14-10
 renaming chunks during 14-16
See also Restoring.
Command-line conventions
 elements of Intro-11
 example diagram Intro-12
 how to read Intro-12
Comment icons Intro-9
Compliance
 with industry standards Intro-16
Components, ON-Bar
 Dynamic Server 2-6
 Extended Parallel Server 2-9
Configuration file
 AC_CONFIG 9-5
 ONCONFIG 9-7
Configuration parameters
 AC_MSGPATH 9-4, 9-5, 9-28
 AC_STORAGE 9-4, 9-5
 AC_VERBOSE 9-4, 9-6
 ALARMPROGRAM 3-11, 4-24, 9-7
 BAR-WORKER_MAX 3-15
 BAR_ACT_LOG 2-17, 3-11, 3-13, 9-8
 BAR_BOOT_DIR 3-13, 9-9
 BAR_BSALIB_PATH 3-9, 3-11, 3-13, 9-10
 BAR_DBS_COSVR 3-13, 9-11
 BAR_HISTORY 3-11, 3-14, 9-12
 BAR_IDLE_TIMEOUT 3-14, 4-32, 9-13
 BAR_LOG_COSVR 3-14, 9-13
 BAR_MAX_BACKUP 3-11, 4-30, 9-14
 BAR_NB_XPORT_COUNT 3-11, 9-15

BAR_PROGRESS_FREQ 3-11, 3-14, 9-16
BAR_RETRY 3-11, 3-14, 9-17
BAR_SM 3-14, 9-19
BAR_SM_NAME 3-14, 9-19
BAR_WORKER_COSVR 3-14, 9-20
BAR_WORKER_MAX 3-17, 9-20
BAR_XFER_BUFSIZE 3-15, 9-23, 9-24
BAR_XFER_BUF_SIZE 3-12, 9-22
BAR_XPORT_COUNT 3-15, 9-24
DBSERVERNAME 10-8
Extended Parallel
 Server 3-12 to 3-16
 global 3-13
ISM_DATA_POOL 3-9, 3-12, 3-15, 9-24
ISM_LOG_POOL 3-9, 3-12, 3-15, 9-25
LTAPEDEV 3-12, 9-26
OFF_RECVRY_THREADS 6-17
ON_RECVRY_THREADS 6-17
RESTARTABLE_RESTORE 6-39, 9-27
 setting, ontape 12-3
 storage-manager section 3-13
Configuring
 ISM 3-4
 third-party storage manager 3-3
Contact information Intro-16
Continuous log backup
 example 2-5
 pausing 8-14
 specifying 1-6, 4-21, 4-27
 using ALARMPROGRAM 9-7
Controlling ON-Bar sessions 8-14
Cooked chunks
 backing up 4-10
 restoring 6-11, 6-15, 6-24
Critical dbspaces, defined 4-5
cron command 2-12
Current log, backup 2-5

Data recovery
 defined 1-3
 See also Restoring.
Data usage 1-16
Database logging
 backups 4-14
 log backups 4-22
Database server
 blocking 7-5, 7-9
 evaluating 1-18 to 3-24
 imported restore 6-32
 migration C-2
 storage-manager
 communication 2-9
 unblocking 7-9
 upgrading 6-32, C-2
 versions with ON-Bar 2-3
Database-logging status, ontape,
 changing 13-5
DBSERVERNAME parameter 5-8, 10-8
Dbslice
 backing up 1-4
 restoring 6-17
Dbspaces
 backing up 1-4
 critical 4-5
 restore selected, ontape 14-4
Default locale Intro-4
Dependencies, software Intro-4
Device, ontape
 /dev/null 13-18
 logical-log backup 13-18
 LTAPEDEV parameter 13-18
 /dev/null, ontape
 as a tape device 12-6
 for logical-log backup 13-18
Disaster recovery, imported
 restore 6-32
Documentation notes Intro-14
Documentation notes, program
 item Intro-15
Documentation, types of Intro-13
 backup and restore tasks 2-4
 documentation notes Intro-14
 machine notes Intro-14
 release notes Intro-14

D

Data consistency, verifying 4-10
Data migration tools C-1

E

Emergency boot file
 backing up 4-6
 how used 2-15
 ixbar 2-15

Environment variables
 AC_CONFIG 5-6, 5-13, 9-5 to 9-6
 GL_DATE D-2
 GL_DATETIME D-2
 INFORMIXSQLHOSTS 10-8
 ISM_DATA_POOL 3-9
 ISM_LOG_POOL 3-9
 typographical conventions Intro-8

en_us.8859-1 locale Intro-4, D-1

Evaluating
 backup and restore time 1-18
 hardware and memory usage 1-18
 logging and transaction activity 1-20, 3-23

Event alarms
 alarmprogram.sh 4-24, 9-7
 setting ALARMPROGRAM 4-24, 9-7

External backup
 blocking database server 2-5, 7-9
 description of 7-5
 procedure 7-7
 tracking backup objects 7-15
 unblocking database server 7-9

External restore
 cold restore procedure 7-19
 examples 2-5, 7-22
 initializing HDR during 7-23
 syntax diagram 7-17
 warm restore procedure 7-21

ex_alarm.sh script 9-7

F

Fake backups 2-5, 4-19

Feature icons Intro-10

Features in 9.3 Intro-6

Files
 emergency boot 2-15
 logical-log 1-5

finderr utility Intro-15, 2-17, 11-3

G

Global Language Support (GLS) Intro-4, D-1

GL_DATE environment variable D-2

GL_DATETIME environment variable D-2

H

Hardware resources, evaluating 1-18

HDR. *See* High-Availability Data Replication.

Help Intro-13

High-Availability Data Replication
 imported restore 6-32
 initializing 6-32 to 6-38

High-Availability Data Replication (HDR)
 initializing with external restore 7-23

I

IBM Informix Server Administrator
 backing up data 4-15
 restoring data 6-15

IBM Informix Storage Manager (ISM)
 backup requests 4-14
 configuring 3-4
 devices supported 3-22
 ISM catalog 4-5, 4-14
 overview 2-13
 requirements 3-20
 sm_versions file 3-5
 upgrading C-2
 Version 2.2 support Intro-3
 volume pool names 3-9

Icons

 feature Intro-10
 Important Intro-9
 platform Intro-10

 product Intro-10
 Tip Intro-9
 Warning Intro-9

Important paragraphs, icon for Intro-9

Importing a restore description 6-6
 initializing HDR 6-38

Incremental backup
 defined 1-17
 example 4-16
 level 1 4-8
 level 2 4-8

Industry standards, compliance with Intro-16

Informix-Admin group 4-3

INFORMIXDIR/bin directory Intro-5

INFORMIXSQLHOSTS environment variable 10-8

Initializing High-Availability Data Replication with ON-Bar 6-36

ISA. *See* IBM Informix Server Administrator.

ISM catalog
 backing up 4-14
 directory path 4-5

ISMData volume pool 3-9

ISMLogs volume pool 3-9

ISM. *See* IBM Informix Storage Manager.

ism_catalog command 4-14, 4-15, 6-10

ism_chk.pl command 6-8

ISM_DATA_POOL parameter 3-9, 3-12, 3-15, 9-24

ISM_LOG_POOL parameter 3-9, 3-12, 3-15, 9-25

ism_startup command 3-5

ism_watch command 4-14, 6-10

ISO 8859-1 code set Intro-4

ixbar.srvvernum file. *See* Emergency boot file.

J

Jukeboxes 3-21

L

Large files, backup
 ON-Bar 4-5
 Level-0 backup 1-4
 Level-1 backup 1-4, 4-8
 Level-2 backup 1-4, 4-8
 Light appends 6-39
 Locale
 description of Intro-4
 using GLS with ON-Bar D-1
 Logging activity, evaluating 1-20, 3-23
 Logical log, ON-Bar
 automatic backup 4-24, 9-7
 blobspace issues 4-19
 checking available space 4-9
 completion messages 4-27
 continuous backup 1-6, 4-21
 current log backup 2-5
 manual backup 4-21, 4-24, 4-26
 replaying records in parallel 6-17
 salvage 2-5
 salvaging 6-19
 skipping replay 4-29
 status of backup 4-21
 when to back up 4-21
 Logical log, ontape
 automatic backup, starting 13-16
 backed-up status 13-15
 backup
 changing parameters 12-10
 device to use 13-18
 if tape fills 13-12
 on another computer 12-6
 procedure 13-14
 separate devices 12-5
 to /dev/null 13-18
 when 13-15
 blobspace blobs 13-14
 continuous backup 13-16
 importance of backing up 1-5
 used status 13-15
 Logical log, overview
 description of 1-5
 manual backup 1-6
 salvaging 1-6
 Logical restore
 description of 1-10, 2-5

ontape
 cold restore 14-4
 warm restore 14-5
See also Restoring.
 Logical-log backup
 current log 7-8
 description 1-5, 4-21 to 4-27
 -O option 4-20
 stopping 4-23
 syntax, IDS 4-22
 syntax, XPS 4-25
 logs_full.sh script 4-24, 9-7
 LTAPEBLK parameter,
 ontape 12-4, 12-7
 LTAPEDEV parameter,
 ON-Bar 3-12, 9-26
 LTAPEDEV parameter, ontape
 changing to /dev/null 12-9
 if two tape devices 13-12
 purpose 13-18
 setting 12-4
 LTAPESIZE parameter 12-4, 12-7, 12-8

M

Machine notes Intro-14
 Manual log backup
 example 4-24, 4-26
 specifying 1-6, 4-21
 Massively parallel-processing
 system 3-23
 Memory resources, evaluating 1-18
 Message file
 usage messages 11-5 to 11-8
See also Activity log, ON-Bar.
 Message file for error
 messages Intro-15
 Migration
 database server C-2
 ontape to ON-Bar C-3
 storage managers C-2
 Mixbar.hostname.servernum
 file 2-16
 Mixed restore
 defined 1-9
 ON-Bar 6-5
 ontape 14-5

MPP system 3-23

O

Offline storage spaces,
 restoring 6-19
 OFF_RECVRVY_THREADS
 parameter 6-17
 ON-Bar
 backup and restore time 1-18
 backup sequence
 Dynamic Server 4-30
 Extended Parallel Server 4-32
 cold restore sequence 6-48, 6-52
 compared to ontape 1-12
 components on
 Dynamic Server 2-6
 Extended Parallel Server 2-9
 configuration
 parameters 3-12 to 3-16,
 9-7 to 9-27
 database servers supported 2-3
 migrating from ontape C-3
 -O option
 backup 4-12, 4-20
 restore 6-11, 6-15, 6-23
 whole-system restore 6-21
 operations supported 1-12
 planning recovery strategy
 creating backup plan 1-17
 data loss 1-14
 data usage 1-16
 failure severity 1-15
 progress feedback 2-17
 renaming chunks 6-27
 starting and stopping
 sessions 8-14
 usage messages 11-5 to 11-8
 warm restore sequence 6-46, 6-50
 where to find information 2-4
 XBSA interface 2-14
See also Configuration parameters;
 Emergency boot file; ON-Bar
 tables; Storage manager.
 ON-Bar activity log. *See* Activity
 log.
 ON-Bar return codes 11-9 to 11-15

onbar script
 description 2-12
 usage and examples 8-3

ON-Bar tables
 bar_action 10-3
 bar_instance 10-5
 bar_object 10-7
 bar_server 10-8
 bar_version 10-8
 described 2-15
 map 10-9, 10-10

onbar-driver
 child process 4-30
 description 2-7, 2-12

onbar_d utility
 purpose 4-32
 See also onbar-driver.

onbar_m utility
 defined 2-12
 ON-Bar diagram 2-11
 purpose 4-32

onbar_w utility
 defined 2-6, 2-12
 ON-Bar diagram 2-11
 purpose 4-32
 starting onbar-workers 8-7

oncfg file 4-6, 4-9

ONCONFIG file. *See* Configuration parameters.

oninit -m command 6-20

Online help Intro-13

Online manuals Intro-13

Online storage spaces,
 restoring 6-23

onmode -c block command 7-9

onmode -c unblock command 7-9

onmode -d command 6-37

onmode -ky command 6-20

onmode -l command 4-19, 4-26

onmode -m command 6-21, 6-40

onmode -sy command 6-21

onmsync utility
 archecker 5-13, 5-14
 defined 2-5
 usage 8-8

onstat command B-1 to B-8

onstat -d command 4-29, 6-8, B-2

onstat -g bus command 4-33, 8-17, B-5

onstat -g bus_sm command 4-33, 8-18, B-5

onstat -l command 4-19, 4-28, B-6

ontape utility
 backing up logical log 13-14
 backup levels 13-6
 backups and modes 13-9
 changing
 database logging status 13-5
 LTAPEDEV to /dev/null 12-9
 parameters 12-10
 TAPEDEV to /dev/null 12-9
 checking configuration
 parameters 12-8
 compared to ON-Bar 1-12
 configuration parameters 12-3
 creating an archive 13-6
 example 13-11
 exit codes 13-4
 if backup terminates
 prematurely 13-9
 labelling backup tapes 13-9
 migrating to ON-Bar C-3
 operations supported 1-12
 option
 -D 14-6
 -L 13-11
 -r 14-6
 -s 13-11
 parameters, changing 12-8
 physical restore, choosing
 type 14-3
 precautions, one tape device 12-5
 read to end of tape 12-7
 restore, choosing mode 14-4
 restoring data 14-3
 starting continuous backup 13-16
 syntax, full backup 13-3
 syntax, logical-log backup 13-16
 writing to end of tape 12-7

ON_RECOVERY_THREADS
 parameter 6-17

Operational table
 backing up 4-21
 restoring 6-39

Override internal checks
 backup 2-5
 restore 2-5

Overriding internal checks
 backup 4-12, 4-20
 restore 6-11, 6-15, 6-23
 whole-system restore 6-21

P

Parallel restore
 example of 2-5

Physical backup 4-7
 description of 4-12
 example of 2-5, 4-20

Physical restore
 description of 1-10
 ON-Bar, example 2-5
 ON-Bar, procedure 6-18
 ontape, cold restore 14-4
 types of 1-11
 See also Restoring.

Planning a backup system 1-18

Platform icons Intro-10

pload utility 4-21

Point-in-log restore 2-5, 6-23

Point-in-time restore
 described 6-6
 example of 2-5, 6-22, D-3

Processes, ON-Bar 4-30,
 4-32 to 6-53

Product icons Intro-10

Program group
 Documentation notes Intro-15
 Release notes Intro-15

Progress, backup or restore 2-17

R

Raw chunks
 backing up 4-10
 restoring 6-24

Raw table
 backing up 4-21
 restoring 6-39

Recovery strategy planning
 creating backup plan 1-17
 data loss 1-14
 data usage 1-16
 failure severity 1-15

- Recovery system
 - comparing ON-Bar and
 - ontape 1-3, 1-12
 - description of 1-3
 - invalid tools 6-18
- Release notes Intro-14
- Release notes, program
 - item Intro-15
- Remote device, ontape
 - and interrupt key 13-17
 - syntax to specify 12-6
 - tape size for 12-8
- Rename chunks restore 2-5
 - described 6-6, 14-5
 - ontape syntax 14-6
 - overview with ON-Bar 6-27
 - overview with ontape 14-16
- Replaying log records 1-7, 6-17
- RESTART option 6-12
- Restartable restore
 - example 6-40
 - example of 2-5
 - overview 6-7
 - usage 6-39
- RESTARTABLE_RESTORE
 - parameter 6-39, 9-27
- Restoring
 - ON-Bar
 - renaming chunks 6-27
- Restoring, ON-Bar
 - cold
 - example of 6-19
 - setting mode 6-4
 - cooked chunks 6-11, 6-15, 6-24
 - db slices 6-17
 - external 2-5, 7-17 to 7-22
 - imported 6-6
 - logical, example 2-5
 - mixed 6-5
 - monitoring progress 2-17
 - O option 2-5, 6-11, 6-15, 6-23
 - offline storage spaces 6-19
 - online storage spaces 6-23
 - operational tables 6-39
 - parallel 2-5
 - physical, example 2-5
 - point-in-log 2-5, 6-23
 - point-in-time example 6-22, D-3
 - raw chunks 6-24

- raw tables 6-39
- rename chunks 2-5
- restartable 2-5, 6-39 to 6-45
- scratch tables 6-38
- selected spaces, example 6-16
- standard tables 6-38
- static tables 6-39
- syntax 5-7, 6-13, 6-28
- table types 6-38
- temp tables 6-38
- using ISA 6-15
- warm 6-16
- whole system 2-5
- Restoring, ontape
 - full-system 14-3
 - selected storage spaces 14-4, 14-14
 - whole system, steps 14-8
- Restoring, overview
 - description of 1-8, 1-10
 - logical phase 1-10, 1-11
 - physical phase 1-10, 1-11
- Return codes 11-9 to 11-15
- Rewindable tape device 12-7
- Rixbar.*hostname.servernum* file 2-16
- Root dbspace, when to back
 - up 4-13, 13-7

S

- Salvaging logs
 - defined 1-6
 - example of 2-5, 6-19
 - skipping 6-19
- Save sets
 - creating 4-11
 - restoring 6-10
- Sbspaces
 - backing up 1-4, 4-18
 - temp space for archecker 5-9
- Scheduling backups 1-19, 2-13
- Scratch table
 - backing up 4-21
 - restoring 6-38
- Server locale D-1
- Session name, assigning to
 - backup 4-20
- Severity of data loss 1-15
- Shared library, XBSA
 - default location 3-9
 - specifying location 3-9, 9-10
- Silos 3-21
- Skipping
 - log salvage 6-19
 - logical replay 4-29
 - storage spaces 4-18
- Smart large objects, backing
 - up 4-18
- SMI tables. *See* Backup Scheduler.
- sm_versions file
 - backing up 4-6, 4-9
 - defining storage manager 3-5
- Software dependencies Intro-4
- sqlhosts file
 - copying 4-6, 4-9
 - server name 10-8
- Standard backup. *See* Backup.
- Standard table
 - backing up 4-21
 - restoring 6-38
- start_worker.sh script 2-12
- Static table
 - backing up 4-21
 - restoring 6-39
- Storage devices
 - backups 5-7, 6-13, 6-28
 - continuous log backups 4-23
 - ISM support 3-22
 - ISM volume pools 4-14, 6-10
 - requirements 3-22
- Storage manager
 - communication with ON-Bar 2-9
 - configurations 3-16 to 3-19
 - migration C-2
 - requirements 3-20, 4-7
 - role in ON-Bar system 2-13
 - sm_versions file 3-5
- Storage spaces
 - backing up a list of 4-16
 - backing up all 4-7, 4-16
 - defined 2-6
 - offline, restoring 6-19
 - online, restoring 6-23
 - physical-only backup 4-20
 - restartable restore 6-40
 - restoring 1-10, 6-16, 6-18
 - skipping during backup 4-18

- when to back up 4-13, 13-7
- stores_demo database Intro-5
- superstores_demo database Intro-5
- Symbolic links, ontape
 - to specify tape devices 12-6
- Syntax diagram
 - backup verification 5-7
 - external backup
 - IDS 7-9
 - XPS 7-10
 - external restore 7-17
 - logical-log backup 4-21, 4-22
 - onbar_w 8-15
 - onsmsync 8-10
 - restore 6-13, 6-28
 - starting and stopping
 - sessions 8-14
 - storage-space backup 4-12
 - summary 4-4
- sysbuobject table 10-12
- sysbuobjses table 10-13
- sysbusession table 10-14
- sysbusm table 10-14
- sysbusmdbspace table 10-15
- sysbusmlog table 10-15
- sysbusmworker table 10-16
- sysbuworker table 10-16
- System requirements
 - database Intro-4
 - software Intro-4
- sysutils database, error
 - messages 9-28

T

- Table
 - altering to
 - raw 6-39
 - standard 4-14
- Table type
 - backing up 4-14, 4-21
 - operational 4-21, 6-39
 - raw 4-21, 6-39
 - restoring 6-38
 - scratch 4-21, 6-38
 - standard 4-21, 6-38
 - static 4-21, 6-39
 - temp 4-21, 6-38

- Tape autochangers 3-21
- Tape device
 - writing to end 12-7
- Tape device, ontape
 - before opening and on
 - closing 12-7
 - block-size parameters 12-7, 12-9
 - gathering for
 - cold restore 14-8
 - warm restore 14-14
 - parameters, setting 12-4
 - precautions with only one 12-5
 - reading to end 12-7
 - rewindable device 12-7
 - specifying symbolic links 12-6
 - using /dev/null 12-6
- Tape libraries 3-21
- Tape stackers 3-21
- TAPEBLK parameter, ontape 12-4, 12-7
- TAPEDEV parameter, ontape
 - changing to /dev/null 12-9
 - described 12-4
 - if two tape devices 13-12
- TAPESIZE parameter, ontape 12-4, 12-7
- Task-documentation matrix 2-4
- Temp table
 - backing up 4-21
 - restoring 6-38
- Temporary spaces 4-4, 4-6
- Text editor, changing ontape
 - parameters 12-10
- Third-party storage manager
 - configuring 3-3
 - functions 2-13
 - onbar script 8-4
- Tip icons Intro-9
- Transaction activity,
 - evaluating 1-20, 3-23

U

- Unblocking, database server 7-9
- UNIX operating system
 - bargroup 3-7, 3-10
 - copying files 7-7
- Upgrading the database server C-2

- Usability enhancements Intro-6
- Utility
 - checker 5-3 to 5-14
 - onsmsync 8-8 to 8-13

V

- Verification, backup 2-5, 5-3 to 5-14
- Virtual processors and ON-
 - Bar 1-18
- Volume pools
 - backup locations 4-14
 - default names 3-9

W

- Warm restore, ON-Bar
 - examples of 6-16
 - sequence 6-46, 6-50
- Warm restore, ontape
 - critical dbspaces 14-15
 - description of 14-5
 - in mixed restore 14-9
 - part of a mixed restore 14-5
 - performing a 14-15
 - steps to perform 14-14
- Warm restore, overview of 1-9
- Warning icons Intro-9
- Whole-system backup
 - defined 2-5
 - overview 4-7
 - specifying 4-13
- Whole-system restore
 - example 2-5, 6-21
 - O option 6-21
 - restartable restore 6-7
 - syntax 6-12, 6-15
- Windows
 - copying files 7-7
 - Informix-Admin group 4-3

X

XBSA interface, described 2-14
XBSA shared library
 default location 3-9
 specifying location 3-9, 9-10
xcfg file 4-6
X/Open compliance level Intro-16

