Informix® Migration Guide

Informix Dynamic Server, Version 7.3
Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options, Version 8.2
Informix Dynamic Server with Universal Data Option, Version 9.1x

Informix Dynamic Server, Developer Edition, Version 7.3
Informix Dynamic Server, Workgroup Edition, Version 7.3

INFORMIX-OnLine, Version 4.1x and 5.x
INFORMIX-OnLine Dynamic Server, Version 6.0 through 7.2x
INFORMIX-OnLine Workgroup Server, Version 7.12 and 7.22
INFORMIX-OnLine XPS, Version 8.1x
INFORMIX-SE, Version 4.1 through 7.2x

February 1998
Part No. 000-4372

Published by INFORMIX® Press

Informix Software, Inc.
4100 Bohannon Drive
Menlo Park, CA 94025-1032

Documentation Team: Twila Booth, Beth Delson, Evelyn Eldridge-Diaz

Table of Contents

Introduction

Section I Migrating Data

Chapter 1 Data Migration Overview

Chapter 2 Moving Between Computers

Section II Migrating Versions of Database Servers

Chapter 3 Migrating OnLine Dynamic Server 7.2 to AD/XP 8.2

Chapter 4 Migrating the 7.2 Database Server to Version 9.1x

Chapter 5 Migrating the Database Server 6.0 or Later to 7.3

Chapter 6 Migrating Pre-6.0 to Informix Dynamic Server 7.3

Chapter 7 Migrating Workgroup Edition and OnLine Workgroup Server

Chapter 9 Moving Between Database Servers

Chapter 10 Changing Locales

Section III Reference

Chapter 11 Utilities for Data Migration

Introduction

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

About This Manual

This manual describes the procedures that you use when you move data from one location to another, as well as instructions for when you migrate between Informix database servers, change locales, or migrate across platforms. *Migration* includes upgrading or reverting to a previous database server version, which often involves changing **sqlhosts** file or registry connectivity information, host environment variables, configuration parameters and other server specific features. This manual describes migrating with the following Informix database servers:

- Informix Dynamic Server
- Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options
- Informix Dynamic Server with Universal Data Option
- Informix Dynamic Server, Developer Edition
- Informix Dynamic Server, Workgroup Edition
- INFORMIX-OnLine Dynamic Server
- INFORMIX-OnLine Workgroup Server
- INFORMIX-OnLine XPS
- INFORMIX-SE
- INFORMIX-Universal Server

This manual also describes how to use the data migration utilities.

Types of Users

This manual is for the following users:

- Database users
- Database administrators
- Database server administrators
- Database-application programmers
- Backup operators
- Performance engineers

This manual assumes that you have the following background:

- A working knowledge of your computer, your operating system, and the utilities provided by your operating system
- Some experience working with relational databases or exposure to database concepts
- Some experience with computer programming

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

This manual assumes that you are using one of the following database servers:

- Informix Dynamic Server, Version 7.3
- Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options, Version 8.2
- Informix Dynamic Server with Universal Data Option, Version 9.1x
- Informix Dynamic Server, Developer Edition, Version 7.3
- Informix Dynamic Server, Workgroup Edition, Version 7.3
- INFORMIX-SE, Version 7.2

Assumptions About Your Locale

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

This manual assumes that you are using the default locale, **en_us.8859-1**. This locale supports U.S. English format conventions for dates, times, and currency. In addition, this locale supports the ISO 8859-1 code set, which includes the ASCII code set plus many 8-bit characters such as é, è, and ñ.

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

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

Demonstration Databases

The DB-Access utility, which is provided with your Informix database server products, includes a demonstration database called **stores7** that contains information about a fictitious wholesale sporting-goods distributor. You can use SQL scripts provided with DB-Access to derive a second database, called **sales_demo**. This database illustrates a dimensional schema for data-warehousing applications. Sample command files are also included for creating and populating these databases.

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

The scripts that you use to install the demonstration databases reside in the **$INFORMIXDIR/bin** directory on UNIX platforms and the **%INFORMIXDIR%\bin** directory on Windows NT platforms. For a complete explanation of how to create and populate the **stores7** demonstration database, refer to the *DB-Access User Manual*. For an explanation of how to create and populate the **sales_demo** database, refer to the *Informix Guide to Database Design and Implementation*.

New Features

The following sections describe new database server features relevant to this manual. For a comprehensive list of new features, see the release notes for your database server.

New Features in Version 7.3

This manual includes information about the following new features:

- Enhanced oncheck options, including index consistency checking without locks
- Built-in server functions
- An administrative interface, Informix Enterprise Command Center (IECC)
- Informix Storage Manager (ISM), a backup and recovery utility
- Optimizer directives

New Features in Version 8.2

This manual describes the following new features that were implemented in Version 8.2 of Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options:

- Enhanced parallel database operations and load balancing across multiple computers
- Enhanced fragmentation of table data on multiple computers
- Enhanced indexes to support data-warehousing applications
- Global Language Support (GLS)
- New **onutil** commands for managing storage spaces
- A new graphical interface, Informix Enterprise Command Center (IECC), for performance monitoring and administration
- Informix Storage Manager (ISM), a native storage manager for backup media

This manual also discusses the following features, which were introduced in Version 8.1 of INFORMIX-OnLine XPS:

- Parallel database operations across multiple computers
- Fragmentation of table data across multiple computers
- Nonlogging tables
- External tables for high-performance loading and unloading
- Coserver groups (cogroups) for centralized administration of coservers
- Dbslices for centralized administration of storage spaces
- The **xctl** and **onutil** utilities

Documentation Conventions

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

The following conventions are covered:

- Typographical conventions
- Icon conventions
- Syntax conventions
- Command-line conventions
- Sample-code conventions

Typographical Conventions

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

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

Tip: *When you are instructed to "enter" characters or to "execute" a command, immediately press* RETURN *after you type the indicated information on your keyboard. When you are instructed to "type" the text or to "press" other keys, you do not need to press* RETURN.

Icon Conventions

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

Comment Icons

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

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

Feature, Product, and Platform Icons

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

Icon	Description
AD/XP	Identifies information that is specific to Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options.
ALS	Identifies information that is specific to an Asian Language Support (ALS) database or application.
D/B	Identifies information that is valid only for DB-Access.
E/C	Identifies information that is specific to the INFORMIX-ESQL/C product.
GLS	Identifies information that relates to the Informix Global Language Support (GLS) feature.
IDS	Identifies information that is specific to Dynamic Server and its editions. However, in some cases, the identified section applies only to Informix Dynamic Server and not to Informix Dynamic Server, Workgroup and Developer Editions. Such information is clearly identified.
IUS	Identifies information that is specific to INFORMIX-Universal Server.
NLS	Identifies information that is specific to a Native Language Support (NLS) database or application.
ODS	Identifies information that is specific to INFORMIX-OnLine Dynamic Server.
OWS	Identifies information that is specific to INFORMIX-OnLine Workgroup Server.
SE	Identifies information that is specific to INFORMIX-SE.
UD	Identifies information that is specific to Informix Dynamic Server with Universal Data Option.

(1 of 2)

Icon	Description
UNIX	Identifies information that is specific to UNIX platforms.
W/D	Identifies information that is specific to Informix Dynamic Server, Workgroup and Developer Editions.
WIN NT	Identifies information that is specific to the Windows NT environment.
XPS	Identifies information that is specific to INFORMIX-OnLine XPS.

(2 of 2)

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

Compliance Icons

Compliance icons indicate paragraphs that provide guidelines for complying with a standard.

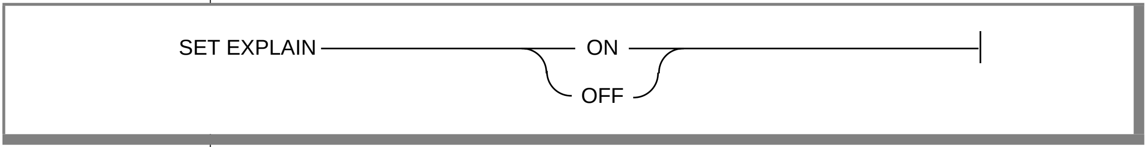
Icon	Description
ANSI	Identifies information that is specific to an ANSI-compliant database.
+	Identifies information that is an Informix extension to ANSI SQL-92 entry-level standard SQL.
X/O	Identifies functionality that conforms to X/Open.

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

Syntax Conventions

This section describes conventions for syntax diagrams. Each diagram displays the sequences of required and optional keywords, terms, and symbols that are valid in a given statement or segment, as Figure 1 shows.

Figure 1
Example of a Simple Syntax Diagram



Each syntax diagram begins at the upper-left corner and ends at the upper-right corner with a vertical terminator. Between these points, any path that does not stop or reverse direction describes a possible form of the statement.

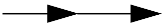
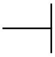
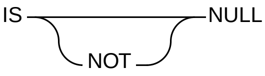
Syntax elements in a path represent terms, keywords, symbols, and segments that can appear in your statement. The path always approaches elements from the left and continues to the right, except in the case of separators in loops. For separators in loops, the path approaches counterclockwise from the right. Unless otherwise noted, at least one blank character separates syntax elements.

Elements That Can Appear on the Path

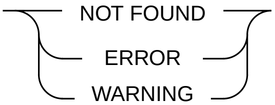
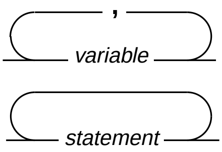
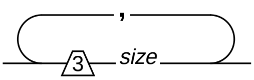
You might encounter one or more of the following elements on a path.

Element	Description
KEYWORD	A word in UPPERCASE letters is a keyword. You must spell the word exactly as shown; however, you can use either uppercase or lowercase letters.
(. , ; @ + * - /)	Punctuation and other nonalphanumeric characters are literal symbols that you must enter exactly as shown.
' '	Single quotes are literal symbols that you must enter as shown.

(1 of 3)

Element	Description
variable	A word in *italics* represents a value that you must supply. A table immediately following the diagram explains the value.
ADD Clause p. 1-14 ADD Clause	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.
Back to ADD Clause p. 1-14	A reference in a box in the upper right hand corner of a subdiagram refers to the next higher-level diagram of which this subdiagram is a member.
Table Name *see* SQLS	A reference to SQLS in this manual refers to the *Informix Guide to SQL: Syntax*. Imagine that the subdiagram is spliced into the main diagram at this point.
VALUES Clause *see* SQLS	A reference to SQLS in this manual refers to the *Informix Guide to SQL: Syntax*. Imagine that the subdiagram is spliced into the main diagram at this point.
Other options *see* OLAG	A reference to OLAG in this manual refers to the administrator's guide for your database server. Imagine that the subdiagram is spliced into the main diagram at this point.
– ALL –	A shaded option is the default action.
	Syntax that is enclosed between a pair of arrows is a subdiagram.
	The vertical line terminates the syntax diagram.
	A branch below the main path indicates an optional path. (Any term on the main path is required, unless a branch can circumvent it.)

(2 of 3)

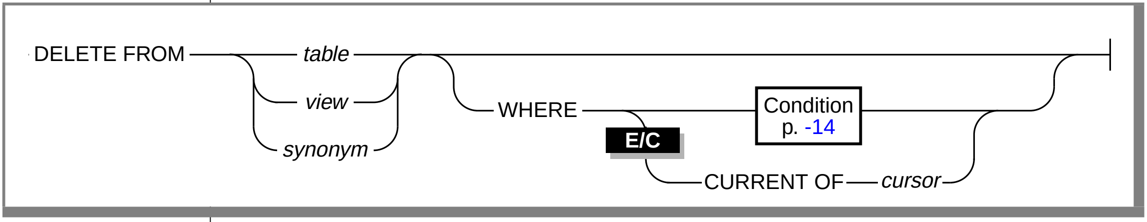
Element	Description
	A set of multiple branches indicates that a choice among more than two different paths is available.
	A loop indicates a path that you can repeat. Punctuation along the top of the loop indicates the separator symbol for list items. If no symbol appears, a blank space is the separator.
	A gate (/3\) on a path indicates that you can only use that path the indicated number of times, even if it is part of a larger loop. Here you can specify *size* no more than three times.

(3 of 3)

How to Read a Syntax Diagram

Figure 2 shows a syntax diagram that uses many of the elements that are listed in the previous table.

Figure 2
Example of a Syntax Diagram



To use this diagram to construct a statement, begin at the far left with the keywords DELETE FROM. Then follow the diagram to the right, proceeding through the options that you want.

To construct a DELETE statement

1. You must type the words `DELETE FROM`.
2. You can delete a table, view, or synonym:
   - Follow the diagram by typing the table name, view name, or synonym, as desired.
   - You can type the keyword `WHERE` to limit the rows that are deleted.
   - If you specify the keyword `WHERE` and you are using DB-Access or the SQL Editor, you must include the Condition clause to specify a condition to delete. To find the syntax for specifying a condition, go to the "Condition" segment on the specified page.
   - If you are using ESQL/C, you can include either the Condition clause to delete a specific condition or the CURRENT OF *cursor* clause to delete a row from the table.
3. Follow the diagram to the terminator. Your DELETE statement is complete.

Command-Line Conventions

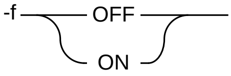
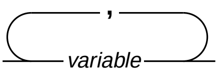
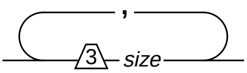
This section defines and illustrates the format of commands that are available in Informix products. These commands have their own conventions, which might include alternative forms of a command, required and optional parts of the command, and so forth.

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

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

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

(1 of 2)

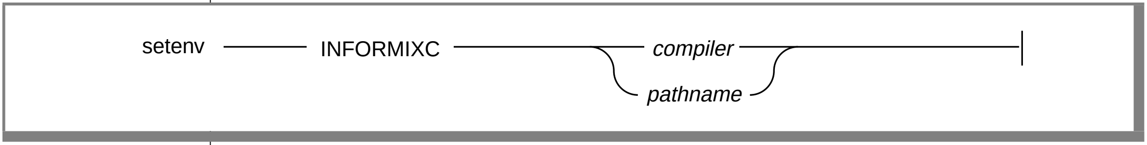
Element	Description
	A branch below the main path indicates an optional path. (Any term on the main path is required, unless a branch can circumvent it.)
	A loop indicates a path that you can repeat. Punctuation along the top of the loop indicates the separator symbol for list items.
	A gate (3) on a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

(2 of 2)

How to Read a Command-Line Diagram

Figure 3 shows a command-line diagram that uses some of the elements that are listed in the previous table.

Figure 3
Example of a Command-Line Diagram



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

Figure 3 diagrams the following steps:

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

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

Sample-Code Conventions

Examples of SQL code occur throughout this manual. Except where noted, the code is not specific to any single Informix application development tool. If only SQL statements are listed in the example, they are not delimited by semicolons. For instance, you might see the code in the following example:

```
CONNECT TO stores7
.
.
.
DELETE FROM customer
    WHERE customer_num = 121
.
.
.
COMMIT WORK
DISCONNECT CURRENT
```

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

Tip: *Ellipsis points in a code example indicate that more code would be added in a full application, but it is not necessary to show it to describe the concept being discussed.*

For detailed directions on using SQL statements for a particular application development tool or SQL API, see the manual for your product.

Additional Documentation

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

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

On-Line Manuals

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

Printed Manuals

To order printed manuals, call 1-800-331-1763 or send email to moreinfo@informix.com. Please provide the following information when you place your order:

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

Error Message Files

Informix software products provide ASCII files that contain all of the Informix error messages and their corrective actions. For a detailed description of these error messages, refer to *Informix Error Messages* in Answers OnLine.

UNIX

To read the error messages in UNIX, you can use the following commands.

Command	Description
finderr	Displays error messages on line
rofferr	Formats error messages for printing

♦

WIN NT

To read error messages and corrective actions in Windows NT, use the **Informix Find Error** utility. To display this utility, choose **Start→Programs→Informix** from the Task Bar. ♦

Documentation Notes, Release Notes, Machine Notes

In addition to printed documentation, the following sections describe the on-line files that supplement the information in this manual. Please examine these files before you begin using your database server. They contain vital information about application and performance issues.

UNIX

On UNIX platforms, the following on-line files appear in the **$INFORMIXDIR/release/en_us/0333** directory.

On-Line File	Purpose
MIGRATEDOC_*x.y*	The documentation-notes file for your version of this manual describes features that are not covered in the manual or that have been modified since publication. Replace *x.y* in the filename with the version number of your database server to derive the name of the documentation-notes file for this manual.
SERVERS_*x.y*	The release-notes file describes feature differences from earlier versions of Informix products and how these differences might affect current products. This file also contains information about any known problems and their workarounds. Replace *x.y* in the filename with the version number of your database server to derive the name of the release-notes file.
IDS_*x.y*, **ONLINE_*x.y***, or **SE_*x.y***	The machine-notes file describes any special actions that are required to configure and use Informix products on your computer. Machine notes are named for the product described. Replace *x.y* in the filename with the version number of your database server to derive the name of the machine-notes file.

♦

WIN NT

The following items appear in the Informix folder. To display this folder, choose **Start→Programs→Informix** from the Task Bar.

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

Machine notes do not apply to Windows NT platforms. ♦

Compliance with Industry Standards

The American National Standards Institute (ANSI) has established a set of industry standards for SQL. Informix SQL-based products are fully compliant with SQL-92 Entry Level (published as ANSI X3.135-1992), which is identical to ISO 9075:1992. 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.

Informix Welcomes Your Comments

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

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

Write to us at the following address:

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

If you prefer to send email, our address is:

doc@informix.com

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

650-926-6571

We appreciate your feedback.

Migrating Data

Chapter

Data Migration Overview

1

This chapter discusses the reasons that you might have for data migration, includes a matrix of the migration paths, and introduces the tools that you can use for migration.

Overview of Informix Database Servers

If you are unfamiliar with the Informix client/server environment, read *Getting Started with Informix Dynamic Server*. It discusses the differences between Informix Dynamic Server and other Informix database servers, and network and server configuration. For information on how to install and configure Informix Dynamic Server with Universal Data Option and its features, read *Getting Started with INFORMIX-Universal Server*. For information on how to install and configure Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options and featured parallel coserver capabilities, see *Getting Started with Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Figure 1-1 lists the Informix database servers by name, latest version level, and the environments they support.

Figure 1-1
Informix Database Servers

Server Type	Version Level	Environments Supported
Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options	8.2	UNIX, WINDOWS NT
Informix Dynamic Server with Universal Data Option	9.13	UNIX, WINDOWS NT

(1 of 2)

Server Type	Version Level	Environments Supported
Informix Dynamic Server	7.3	UNIX, WINDOWS NT
Dynamic Server, Workgroup and Developer Editions	7.3	WINDOWS NT, UNIX
INFORMIX-SE	7.22	UNIX, WINDOWS NT

(2 of 2)

For installation instructions, refer to the *Installation Guide* for your database server. For information on how to administer and configure your database server, refer to your *Administrator's Guide*.

Planning for Data Migration

You might need to perform data migration on a database or selected data for the following reasons:

- Migrating database server versions
- Changing database servers or environments
- Distributing a client application
- Importing non-Informix data

Migrating Database Server Versions

Consider changes in the configuration parameters and environment variables, the amount of memory and dbspace space they require, and the organization of the data.

Changing Database Servers or Environments

You might want to move between database servers, to a different operating system, or to a different locale. If you are moving to a different operating system, Dynamic Server with AD and XP Options, or INFORMIX-SE, you must unload the data into text files and then load it back again into the target database server.

You might want to change the database schema to accommodate more information, to provide for growth, or to enhance performance.

Distributing a Client Application

Verify that the client application version you use is compatible with your database server version. Update the **sqlhosts** file or registry information for the client applications with the new database server information.

After you upgrade a database server on the same operating-system platform, or move the database server to another compatible computer, review the client applications and **sqlhosts** file or registry connections. You might need to recompile or modify client applications, or update the **sqlhosts** file or registry information.

For more information about interactions between client applications and different database servers, refer to the client manual, such as the *INFORMIX-ESQL/C Programmer's Manual* or the *DataBlade API Programmer's Manual*.

Importing Non-Informix Data

You can use the **dbimport** and **dbload** utilities, the High-Performance Loader (HPL), the INFORMIX-Enterprise Gateway products, or external tables for Dynamic Server with AD and XP Options, to import data from non-Informix sources. For more information, see "Importing Data from a Non-Informix Source" on page 2-9.

Informix Database Server Versions

Figure 1-2 lists all the database server versions that Informix supports on UNIX. You might have previously purchased other Informix database server versions. If you have a database server version not listed here, you can migrate to a version that is supported.

To migrate between database server versions marked with a pound sign (#), you must migrate through Version 7.2. To migrate between UNIX and Windows NT, see Chapter 9, "Moving Between Database Servers."

Figure 1-2
Migrating Between Database Servers on UNIX

	To Version																	
From Version	**9.13.U**	**9.12.U**	**9.11.U**	**9.10.U**	**8.11.U**	**8.10.U**	**7.30.U**	**7.24.U**	**7.23.U**	**7.22.U**	**7.21.U**	**7.20.U**	**7.14.U**	**7.13.U**	**7.12.U**	**7.11.U**	**7.10.U**	**5.0x.U**
9.13.U	●																	
9.12.U	●	●																
9.11.U	●	●	●															
9.10.U	●	●	●	●														
8.11.U					●													
8.10.U					●	●												
7.30.U	●	●	●	●	●	●	●											
7.24.U	●	●	●	●	●	●	●	●										
7.23.U	●	●	●	●	●	●	●	●	●									
7.22.U	●	●	●	●	●	●	●	●	●	●								
7.21.U	●	●	●	●	●	●	●	●	●	●	●							
7.20.U	●	●	●	●	●	●	●	●	●	●	●	●						
7.14.U	#	#	#	#	#	#	#	●	●	●	●	●	●					
7.13.U	#	#	#	#	#	#	#	●	●	●	●	●	●	●				
7.12.U	#	#	#	#	#	#	#	●	●	●	●	●	●	●	●			
7.11.U	#	#	#	#	#	#	●	●	●	●	●	●	●	●	●	●		
7.10.U	#	#	#	#	#	#	●	●	●	●	●	●	●	●	●	●	●	
5.0x.U							#	●	●	●	●	●	●	●	●	●	●	●

Figure 1-3 lists all the database server versions Informix supports on Windows NT.

Figure 1-3
Migrating Between Database Servers on Windows NT

	To Version						
From Version	**9.12.T**	**7.30.T**	**7.23.T**	**7.22.T**	**7.20.T**	**7.12.T**	**7.10.T**
9.12.T	●						
7.30.T		●					
7.23.T		●	●				
7.22.T		●	●	●			
7.20.T		●	●	●	●		
7.12.T					●	●	
7.10.T						●	●

Database Server Migration Paths

This section summarizes the migration paths for Informix database servers. Figure 1-4 shows where you can find information in this book on how to upgrade to another Informix database server. Figure 1-5 shows where you can find information in this book on how to revert to an earlier version of an Informix database server. The **Source** column shows the original database server and environment. The **Target** column shows the new database server and environment to which you are migrating. The **Page Reference** column shows where you can find more information about the specific migration path.

Important: *To update your database server, you must first perform a clean shutdown. Wait for all activity to cease before you proceed. Perform a level-0 backup as soon as the new database server is on-line.*

Figure 1-4 through Figure 1-7 use the following abbreviations for the database server names.

Database Server Name	Abbreviation
Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options	AD/XP
INFORMIX-Universal Server	IUS
Informix Dynamic Server	IDS
INFORMIX-OnLine Dynamic Server	ODS
INFORMIX-OnLine Workgroup Server	OWS
INFORMIX-SE	SE
Dynamic Server, Workgroup and Developer Editions	W/D
Universal Data Option	UD
INFORMIX-OnLine XPS	XPS

Figure 1-4
Upgrading Database Servers

Source	Target	Page Reference
AD/XP on UNIX or Windows NT 8.1	AD/XP on UNIX or Windows NT 8.2	3-42
ODS on UNIX or Windows NT 7.2	AD/XP on UNIX or Windows NT 8.2	3-22
ODS on UNIX or Windows NT 7.2	IUS or UD	4-9
ODS on UNIX 6.0	IDS on UNIX 7.3	5-23
ODS on UNIX (pre-6.0)	IDS on UNIX 7.3	6-24
ODS on Windows NT 7.12	IDS on Windows NT 7.3	5-23
OWS on Windows NT 7.12	OWS on Windows NT 7.22	7-10
OWS on UNIX 7.12	OWS on UNIX 7.3	7-10
SE (any earlier version)	SE 7.2x	8-3

Figure 1-5
Reverting Database Servers

Source	Target	Page Reference
AD/XP on UNIX or Windows NT 8.2	XPS on UNIX or Windows NT 8.1	3-42
AD/XP on UNIX or Windows NT 8.2	IDS on UNIX or Windows NT 7.3	3-41
UD on UNIX or Windows NT 9.13	IUS on UNIX or Windows NT 9.1	4-30
UD on UNIX or Windows NT 9.13	ODS on UNIX or Windows NT 7.2	4-22
IDS on UNIX or Windows NT 7.3	ODS on UNIX or Windows NT 6.0 or later	5-33
ODS on UNIX 7.2x	ODS on UNIX (pre-6.0)	6-39
ODS on Windows NT 7.22	ODS on Windows NT 7.12 or 7.2	5-33
OWS on UNIX 7.22	OWS on UNIX 7.12	7-21
W/D on Windows NT 7.3	OWS on Windows NT 7.12	7-21
SE 7.2x	SE (any earlier version)	8-6

Automatic Migration Between Database Servers

Figure 1-6 shows situations where data migration is automatic. Automatic data migration means you do not need to use **dbexport**, UNLOAD, **onunload**, the High-Performance Loader (HPL), or external tables, to move the data. Simply install and start the new database server. For example, data migration between OnLine Dynamic Server (ODS) and OnLine Workgroup Server (OWS) is automatic if the database servers are on the same operating system.

Data migration is also automatic when you move between different versions of a database server in the same operating system.

Figure 1-6
Migrating Automatically Between Different Database Servers

Source	Target	Page Reference
ODS, IDS	OWS, W/D	9-16
OWS, W/D	ODS, IDS	9-26

Migrating to a Different Operating System

Figure 1-7 shows all the paths for migrating to a different operating system. For example, if you plan to migrate from SE to OnLine Workgroup Server, refer to "Moving Between Database Servers" on page 9-1.

Figure 1-7
Migrating to a Different Database Server

Source	Target	Page Reference
AD/XP on UNIX or WINDOWS NT	AD/XP on UNIX or WINDOWS NT	9-5
IUS or UD on UNIX or WINDOWS NT	IUS or UD on UNIX or WINDOWS NT	9-5
C-ISAM	SE, ODS, IDS, OWS, or W/D	8-6
ODS, IDS on WINDOWS NT	ODS, IDS on UNIX	9-36
ODS, IDS on WINDOWS NT	OWS, W/D on UNIX	9-16
ODS, IDS on UNIX or WINDOWS NT	SE (all environments)	9-11
ODS, IDS on UNIX	ODS, IDS on WINDOWS NT	9-36
ODS, IDS on UNIX	OWS, W/D on WINDOWS NT	9-16
OWS, W/D on WINDOWS NT	ODS, IDS on UNIX	9-26
OWS, W/D on WINDOWS NT	OWS, W/D on UNIX	9-38
OWS, W/D on WINDOWS NT or UNIX	SE (all environments)	9-11
OWS, W/D on UNIX	ODS, IDS on WINDOWS NT	9-16
OWS, W/D on UNIX	OWS, W/D on WINDOWS NT	9-38
SE (all environments)	ODS, IDS or OWS, W/D on UNIX OR WINDOWS NT	9-6

Tools for Moving Data

You can use the following tools, utilities, and SQL statements to move data from one database to another:

- The **onunload** and **onload** utilities (only between database servers of the same version)
- The **dbexport** and **dbimport** utilities
- The LOAD and UNLOAD statements
- The **dbload** utility
- External tables
- The Informix Enterprise Command Center
- High-Performance Loader (HPL)

Figure 1-8 lists which tools to use for each database server.

Figure 1-8
Utilities for Moving Data

Utility	Database Server						
	XPS	**AD/XP**	**IDS**	**OWS** or **W/D**	**ODS**	**IUS**	**UD**
onunload / onload	●	●	●	●	●	●	●
dbexport / dbimport			●	●	●	●	●
LOAD / UNLOAD	●	●	●	●	●	●	●
dbload			●	●	●	●	●
External tables	●	●					
IECC	●	●	●	●	●		
High-Performance Loader (HPL)			●		●	●	●

The best method for moving data depends on your operating system and whether you want to move an entire database, selected tables, or selected columns from a table. Figure 1-9 summarizes the characteristics of the utilities that load data and the constraints and advantages of each method.

Figure 1-9
Comparison of Tools for Loading Data

	onload/ onunload	dbexport/ dbimport	UNLOAD/ LOAD	dbload	HPL	External Tables	IECC
Granularity of data	Table or database	Database only	Partial or complete table	Partial or complete table	Partial or complete table	Partial or complete table	Tables or database
Performance	Fast	Moderate	Moderate	Slow	Very fast	Fast	Moderate
Source of data	Must be produced by **onunload**	Usually produced by **dbexport**	Any data in the specified format, usually produced by UNLOAD	Any data in the format specified by the input file	Any ASCII or COBOL data. User can create custom read capabilities.	Any data in the format specified by the input file	Data must be unloaded by IECC
Database schema	Cannot modify	Can modify	Can modify	Can modify	Can modify	Can modify	Cannot modify
Location of data	Disk or tape	Disk or tape	Disk only	Disk only	Disk, tape, or pipe	Disk, tape, or pipe	Disk or tape
Type of file	Binary	Text	Text	Text	Text	Text	Text
Logging status	Logging must be turned off	Logging optional	Logging optional	Logging optional	Logging optional	Logging optional	Logging optional
Environment	Cannot use when you move between environments	Can use when you move between environments	Can use when you move between environments	Can use when you move between environments	Can use when you move between environments or from a non-Informix database	Can use when you move between environments	Can use when you move between environments
Ease of use	More difficult	Moderate	Easiest	Moderate	Most difficult	Most difficult	Easy to use

Constraints of Migration Tools

The **onunload** and **onload** utilities provide the fastest way to move data, but they do not let you modify the database schema or move from one platform or version to another. For more information on **onunload** and **onload**, see Figure 1-10 on page 1-15. The **dbexport** and **dbimport** utilities provide some flexibility, but you must move an entire database. The **dbload** utility gives you a great deal of flexibility, but it is not as fast as the other methods, and you must prepare a command file to control the input. You can use **dbload** with data in a variety of formats. The LOAD statement is moderately fast and easy to use, but it can only accept specified data formats. You usually use LOAD with data that is prepared with an UNLOAD statement. External tables are only used in Dynamic Server with AD and XP Options. For Informix Dynamic Server, Version 7.3, you can import and export a database or unload and load tables with Informix Enterprise Command Center. For more information on Informix Enterprise Command Center, see the *INFORMIX-Enterprise Command Center User Guide* for Informix Dynamic Server.

For more information on these utilities, see Chapter 11, “Utilities for Data Migration.”

Choosing the Appropriate Migration Utility

This section provides guidelines for how to choose the appropriate migration utility.

IDS
IUS
ODS
UDS

Using the High-Performance Loader

The High-Performance Loader (HPL) requires significant preparation time but is fast. Use the HPL for large migration jobs. The HPL can load data from any ASCII or COBOL file that meets certain format requirements. Only Informix Dynamic Server or OnLine Dynamic Server for UNIX, Version 7.2 or later, and Universal Server or Dynamic Server with UD Option support the HPL.

OnLine Dynamic Server for Windows NT; OnLine Workgroup Server; Informix Dynamic Server, Workgroup and Developer Editions; OnLine XPS; and Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options do not support the HPL.

For information about how to use the HPL, refer to the *Guide to the High-Performance Loader*.

AD/XP

XPS

Using External Tables

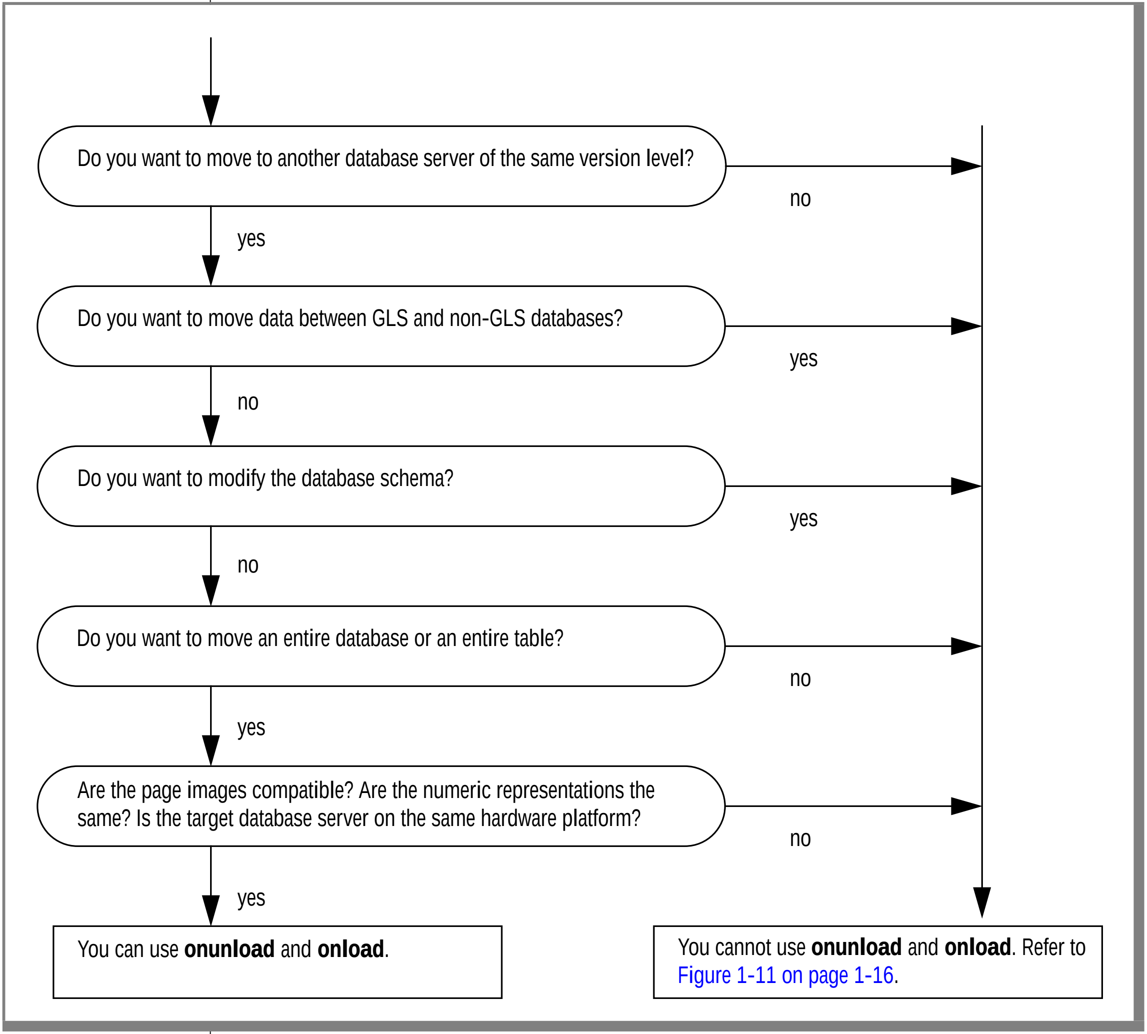
Only INFORMIX-OnLine XPS and Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options use external tables to load and unload data. For more information on how to load data into INFORMIX-OnLine XPS and Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options, see Chapter 3, "Migrating OnLine Dynamic Server 7.2 to AD/XP 8.2."

Using the onunload and onload Utilities

The **onunload** utility unloads data from the specified database or table onto a tape or a file on disk in disk-page-sized units. The **onload** utility takes a tape or a file that the **onunload** utility creates and re-creates the database or the table. The **onunload** and **onload** utilities are faster than the **dbimport**, **dbload**, or LOAD options but are much less flexible.

Because the data is written in page-sized units, you can use **onunload** and **onload** only when certain constraints are met. For example, you cannot use **onunload** and **onload** to move data between UNIX and Windows NT. These constraints are discussed in "Constraints That Affect onunload" on page 11-74. Figure 1-10 summarizes the questions that you must ask before you use **onunload** and **onload**.

Figure 1-10
Criteria for Using the onunload and onload Utilities


Do you want to move to another database server of the same version level?
no
yes
Do you want to move data between GLS and non-GLS databases?
yes
no
Do you want to modify the database schema?
yes
no
Do you want to move an entire database or an entire table?
no
yes
Are the page images compatible? Are the numeric representations the same? Is the target database server on the same hardware platform?
no
yes
You can use **onunload** and **onload**.
You cannot use **onunload** and **onload**. Refer to Figure 1-11 on page 1-16.

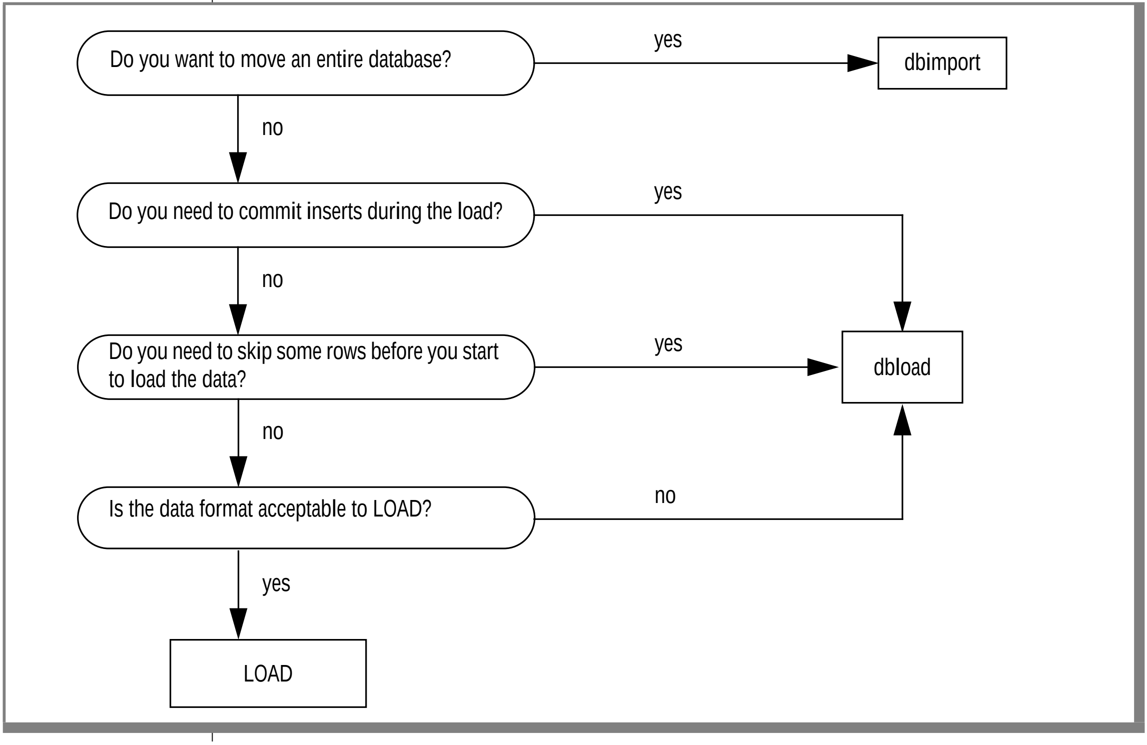
Using dbload, dbimport, and LOAD

If you cannot use **onunload** and **onload**, you must choose between the **dbload** utility (to load), the **dbexport** and **dbimport** utilities, and the LOAD and UNLOAD SQL statements. All these methods allow you to modify the database schema.

The **dbimport** utility loads a complete database. Use **dbimport** when you must move a complete database and cannot use **onload**. To load tables, use LOAD or **dbload**.

To manipulate the data file you are loading, or access the database while it is loading, use the **dbload** utility. The cost of the flexibility is the time you spend creating the **dbload** command file and slower execution. When possible, use the LOAD statement. The LOAD statement is faster than **dbload**. Figure 1-11 summarizes questions to help you choose among these methods.

Figure 1-11
Choosing Among dbimport, dbload, and LOAD


Do you want to move an entire database?
yes
dbimport
no
Do you need to commit inserts during the load?
yes
no
Do you need to skip some rows before you start to load the data?
yes
dbload
no
Is the data format acceptable to LOAD?
no
yes
LOAD

Chapter

Moving Between Computers

2

This chapter discusses how to move data between different computers and how to import data from non-Informix environments. Except when you use the High-Performance Loader (HPL) or external tables, you must unload your data to ASCII files before you move the data to another computer.

Using onunload and onload to Migrate Data

The **onunload** and **onload** utilities are the fastest way to unload and load data, but you can use them only when the source and target computers have the same:

- page size.
- representation of numeric data.
- byte alignment for structures and unions.
- database server version.

You cannot use **onunload** and **onload** to move data between UNIX and Windows NT because they use different page sizes. For example, the page size is 2 kilobytes on some UNIX systems and 4 kilobytes on Windows NT.

You can use the **onunload** and **onload** utilities to unload and load data from the following database servers between compatible computers:

Database Server Data	Computer Platform
Informix Dynamic Server	UNIX
Informix Dynamic Server	Windows NT
OnLine Dynamic Server	UNIX
OnLine Dynamic Server	Windows NT
Informix Dynamic Server, Workgroup and Developer Editions	UNIX
Informix Dynamic Server, Workgroup and Developer Editions	Windows NT
OnLine Workgroup Server	UNIX
OnLine Workgroup Server	Windows NT
Universal Server	UNIX (only if the database does not contain extended data types)
Universal Server	Windows NT (only if the database does not contain extended data types)
Informix Dynamic Server with Universal Data Option	UNIX (only if the database does not contain extended data types)
Informix Dynamic Server with Universal Data Option	Windows NT (only if the database does not contain extended data types)

For example, your site purchases a more powerful UNIX computer to allow faster access to the users. You need to transfer existing databases to the new database server on the new computer. Use **onunload** to unload data from the first database server, then use **onload** to load the data into the second database server. Both database servers must be at the same version level. You can move the entire database or selected tables only, but you cannot modify the database schema.

The **onunload** and **onload** utilities have the following restrictions:

- You cannot use **onunload** and **onload** to move data between non-GLS and GLS locales.

IUS

UD

- Do not use **onunload** and **onload** to move data between two Universal Server or Informix Dynamic Server with Universal Data Option databases if they contain extended data types.

  Use the HPL instead to move the Universal Server data. However, you can use **onunload** and **onload** with Universal Server if the databases contain legacy data types. ♦

SE

- INFORMIX-SE does not support **onload** and **onunload**. ♦

AD/XP

XPS

- INFORMIX-OnLine XPS and Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options do not support **onload** and **onunload**.

  Use external tables to unload and load your data when you use INFORMIX-OnLine XPS or Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options to move data between compatible computers. For information on how to load data with OnLine XPS, see Chapter 3, “Migrating OnLine Dynamic Server 7.2 to AD / XP 8.2.” ♦

Steps for Using onunload and onload

This section describes the procedure for how to use **onunload** and **onload** to move a database. You can use these commands to move either a complete database or a table from one computer to another. The syntax and description of the **onunload** utility starts on page 11-73. The syntax and description of the **onload** utility starts on page 11-62.

To move a database from one computer to another

1. Make sure that the page size, numeric representations, and byte alignment on structures and unions are the same on both computers.

   (The page size is 2 kilobytes on certain UNIX systems and 4 kilobytes on Windows NT.) The page size is an Informix characteristic. For information about page size, refer to your *Administrator's Guide*. The numeric representation and the byte alignment are characteristics of your operating system. For information about numeric representation and byte alignment, refer to the manuals for your operating systems.

2. Decide where to store the unloaded data.

   On disk — Create an empty file for **onunload** to hold the data. Make sure that you have write permission for the file.

   On tape — Use the tape device and characteristics specified in the ONCONFIG configuration file by either TAPEDEV or LTAPEDEV, or specify another tape device. Make sure that the tape device that you specify is available for **onunload.**

3. Run the **oncheck** utility to make sure that your database is consistent.

   For information about **oncheck**, refer to your *Administrator's Guide*.

4. If you want to save the triggers, access privileges, stored procedures, defaults, constraints, and synonyms for the tables in the database, run the **dbschema** utility.

5. Run the **onunload** utility.

6. If necessary, transfer the storage medium (tape or disk) to the new computer.

   If the two computers are on the same network, you can read or write the data remotely.

7. Run the **onload** utility.

8. Set the desired logging status for the new database.

   For information about logging status, refer to your *Administrator's Guide*.

9. If necessary, change the DBA privileges of the database.
10. If you want to restore the triggers, access privileges, stored procedures, defaults, constraints, and synonyms for the tables in the database, run the **dbschema** utility.
11. Create a level-0 backup of the new database.

To move a table from one computer to another

1. Make sure that the page size, numeric representations, and byte alignment on structures and unions are the same on both computers. (The page size is 2 kilobytes on certain UNIX systems and 4 kilobytes on Windows NT.)
2. Decide where to store the unloaded data. (See step 2 of the previous section.)
3. Run the **oncheck** utility to make sure that your database is consistent.
4. If you want to save the triggers, access privileges, stored procedures, defaults, constraints, and synonyms for the table, run the **dbschema** utility.
5. Run the **onunload** utility.
6. If necessary, transfer the storage medium to the new computer.
7. If the table includes simple large objects that are stored in blobspaces, decide where to store the simple large objects. If necessary, create new blobspaces.
8. Turn off logging.

   When you are loading a table, logging on the target database must be turned off. (When you are creating and loading an entire database, the logging status does not matter.)
9. Run the **onload** utility.
10. Create a level-0 backup of the modified database.
11. Turn logging back on, if you want logging.
12. If you want to restore the triggers, access privileges, stored procedures, defaults, constraints, and synonyms for the table, run the **dbschema** utility or create them manually.

To move a table from one dbspace to another dbspace on the same computer

1. Run the **onunload** utility to unload the table.
2. Turn off logging.

   When you are loading a table, logging on the target database must be turned off.
3. Run the **onload** utility.

   Specify a new table name and new dbspace name in the **onload** statement.
4. If the data loads successfully, delete the old table in the old dbspace and rename the new table to the old table name.
5. Create a level-0 backup of the modified database.
6. Turn logging back on, if you want logging.

Other Migration Utilities

If you cannot use the **onunload** and **onload** utilities to export and import data, you must unload your data to text files. You can use the **dbexport** utility to unload Universal Server, Dynamic Server with UD Option, Informix Dynamic Server, OnLine Dynamic Server, OnLine Workgroup Server, Dynamic Server with AD and XP Options, and SE data to tape. The UNLOAD statement lets you manipulate the data as you unload it, but it requires that you unload to files on disk instead of to tape. If you unload to files, you might need to use UNIX or Windows NT utilities to load those files onto tape.

If you are moving to an Informix database server on another computer, you can use the **dbimport** and **dbload** utilities to load the data that you exported.

If you are moving data to a non-Informix application, you might need to use the UNLOAD statement because it lets you specify the delimiter that is used in the data files.

For more information on the **dbexport**, **dbimport**, **dbload**, and **dbschema** utilities, see Chapter 11, "Utilities for Data Migration." For more information about the UNLOAD and LOAD statements or external tables, refer to the *Informix Guide to SQL: Syntax*.

Importing Data from a Non-Informix Source

The **dbimport** and **dbload** utilities can import data from any ASCII file that is properly formatted. Most applications that produce data can export the data into files that have a suitable format for **dbimport**. If the format of the data is not suitable, use UNIX or Windows NT utilities to reformat the data before you import it into Universal Server, OnLine Dynamic Server, OnLine Workgroup Server, or SE.

In addition to **dbimport** and **dbload**, the INFORMIX-Enterprise Gateway products and the High-Performance Loader (HPL) provide ways to access information from non-Informix sources.

Using the INFORMIX-Enterprise Gateway Products

INFORMIX-Enterprise Gateway with DRDA lets you query databases that conform to the DRDA protocol published by IBM. You can use this Gateway product to query a DRDA database and then insert the results into an Informix database. For example, to import data, execute a SELECT statement from the non-Informix database and then an INSERT statement into the Informix database. For more information about INFORMIX-Enterprise Gateway with DRDA, refer to the *INFORMIX-Enterprise Gateway with DRDA User Manual*.

INFORMIX-Enterprise Gateway for EDA/SQL lets you issue queries on a variety of hardware platforms. It accesses an EDA/SQL server from Information Builders, Inc. (IBI), which in turn accesses the data source. For more information about INFORMIX-Enterprise Gateway for EDA/SQL, refer to the *INFORMIX-Enterprise Gateway for EDA/SQL User Manual*.

UNIX

INFORMIX-Enterprise Gateway provides a single, standards-based gateway to multiple data sources. Gateway Manager connects the Informix environment with that of any shared-library ODBC Level 2-compliant driver manager and driver(s) on UNIX. For instance, you can use Gateway Manager with the INFORMIX-Enterprise Gateway driver products to access UNIX database server products such as SYBASE SQL Server 10 and ORACLE7 Server. For more information about Gateway Manager, refer to the *INFORMIX-Enterprise Gateway Manager User Manual*. ♦

IDS
IUS
ODS
UD

Using the High-Performance Loader

High-Performance Loader (HPL) uses parallel processing to perform fast data loading and unloading. The High-Performance Loader (HPL) is a utility available with the following servers:

- Informix Dynamic Server
- OnLine Dynamic Server
- Universal Server
- Dynamic Server with UD Option

Only the UNIX versions of Universal Server, Dynamic Server with UD Option, OnLine Dynamic Server, and Informix Dynamic Server, Version 7.2x and higher, support the HPL. The Windows NT version of OnLine Dynamic Server and the UNIX and Windows NT versions of OnLine Workgroup Server do not support the HPL.

You can use the HPL to load from large ASCII or COBOL databases. COBOL is supported up to Version 7.3.

In addition to the advantage of its speed, the following HPL features provide powerful tools for handling data from non-Informix sources:

- Drivers to handle different database types
- Filters and functions to manipulate data
- Code-set conversion

For more information about the HPL, refer to the *Guide to the High-Performance Loader*.

Section II

Migrating Versions of Database Servers

Chapter

3

Migrating OnLine Dynamic Server 7.2 to AD/XP 8.2

This chapter describes the procedure to migrate to Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options, Version 8.2, from OnLine Dynamic Server, Version 7.2.

This chapter covers the following topics:

- Preparing to migrate to Dynamic Server with AD and XP Options
- Upgrading OnLine Dynamic Server, Version 7.2, to Dynamic Server with AD and XP Options, Version 8.2
- Reverting Dynamic Server with AD and XP Options to OnLine Dynamic Server, Version 7.2
- Upgrading OnLine XPS 8.1 to Dynamic Server with AD and XP Options, Version 8.2.

Tip: *To migrate from Version 7.3 to Dynamic Server with AD and XP Options, revert to Version 7.2 first.*

Preparing to Migrate to Dynamic Server with AD and XP Options 8.2

Informix suggests that you observe the following precautions when you migrate to Dynamic Server with AD and XP Options, Version 8.2:

- Check the release notes for information about the proper operating-system release and any patches that you require for successful installation and operation of the database server.

  The release notes are in one of the following directories:

  UNIX

  - ❑ **$INFORMIXDIR/release/en_us/0333**♦

  WIN NT

  - ❑ **%INFORMIXDIR%\release\en_us\0333.**

    Release Notes appear in the Informix folder. To display this folder, choose **Start→Programs→Informix** from the Task Bar. ♦

  On UNIX, retain both versions of the Informix product software on disk (if you have enough disk resources). On Windows NT, you cannot retain two versions of the Informix product on disk.

- Retain the installation tapes for UNIX or the CDs for Windows NT from both versions of the Informix product software.
- Perform a level-0 backup of all dbspaces and blobspaces with OnLine Dynamic Server, Version 7.2.

  After you complete the migration, perform another level-0 backup with Dynamic Server with AD and XP Options.

- Use a test instance of Dynamic Server with AD and XP Options to test the installation and migration procedures.

  Use a test instance in the desired communication mode to practice bringing the new database server on-line before you attempt to convert the production database.

For additional information, refer to the *Installation Guide* for your database server.

Changes Introduced by Dynamic Server with AD and XP Options 8.2

This section describes the changes that Dynamic Server with AD and XP Options introduces that affect initial configuration and migration. Dynamic Server with AD and XP Options introduces new features and associated terminology including coservers, cogroups, and dbslices that facilitate a high degree of parallelism.

High Degree of Parallelism

This high degree of parallelism allows SQL operations to use multiple threads simultaneously on all central processing unit (CPU) virtual processors (VPs) and across all coservers to speed up execution of any single query. Parallel execution occurs automatically and is optimized when data is fragmented across multiple dbspaces and multiple CPU VPs are available. Parallel processing can occur on a single coserver or across multiple coservers. To support parallelism and the new features of Dynamic Server with AD and XP Options, additional configuration parameters and new environment variables are available.

What are Coservers, Cogroups, and Dbslices?

You use coservers, cogroups, and dbslices for multiple-coserver confirmation. If you have a single coserver configuration, you can skip the discussion of multiple coservers and go to "Environment Variables" on page 3-6.

Coservers and Cogroups

A Dynamic Server with AD and XP Options *coserver* is the functional equivalent of an instance of OnLine Dynamic Server or Informix Dynamic Server. Dynamic Server with AD and XP Options coservers execute on different nodes. Usually one coserver exists per node. Coservers collectively communicate with each other to provide a single more powerful Dynamic Server with AD and XP Options database server. Coserver numbers start from 1 and continue to the number of nodes configured on your system.

A *cogroup* is a subset, or the entire set, of the coservers within a Dynamic Server with AD and XP Options database. Cogroups are object names to a number of coservers on your system. Cogroups can help you administer a larger number of coservers.

Dbslices

A *dbspace* is a logical storage unit that allows you to allocate data strategically. With dbspaces, you can link specific physical units (such as chunks) and logical units (such as tables) of a database. To manage many dbspaces, Dynamic Server with AD and XP Options uses the concept of a *dbslice*, which is a named set of dbspaces. You can manage a dbslice as a single storage object.

A dbslice simplifies management of storage spaces because you can use the dbslice name to refer to all of the storage spaces for a single table.

For more information on coservers, cogroups, and dbslices, see *Getting Started with Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options* and the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Environment Variables

Dynamic Server with AD and XP Options introduces significant changes to the following environment variables:

- INFORMIXSERVER
- PDQPRIORITY
- PSORT_NPROCS

The *Informix Guide to SQL: Reference* describes these environment variables. For performance implications and guidelines, refer to the *Performance Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Configuration Parameters

Dynamic Server with AD and XP Options, Version 8.2, introduces new configuration parameters that might affect installation. You might also need to adjust the values of existing parameters. Figure 3-1 lists the ONCONFIG parameters for a single-coserver configuration for Dynamic Server with AD and XP Options, Version 8.2. Figure 3-2 on page 3-9 lists the ONCONFIG parameters for a multiple-coserver configuration for Dynamic Server with AD and XP Options, Version 8.2.

For more information on these configuration parameters, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Single Coserver

For a single-coserver configuration, you can use the default values in the **onconfig.std** template that is located in the **etc** directory in **INFORMIXDIR.**

Important: *Do not modify* ***onconfig.std****. The database server provides the* ***onconfig.std*** *configuration file as a template and not as a functional configuration.*

For more information on template configuration files, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Figure 3-1
Single-Coserver Configuration Parameters

Parameters Added	Parameters Changed	Parameters Dropped
CONFIGSIZE	MIRRORPATH	DS_MAX_SCANS
PHYSDBS	ROOTNAME	
ROOTNAME	ROOTPATH	
ISO_CURLOCKS	DBTEMPSPACE	
DS_ADM_POLICY	DS_TOTAL_MEMORY	
BAR_ACT_LOG	MAX_PDQPRIORITY	
BAR_RETRY	DBSERVERNAME	

(1 of 2)

Parameters Added	Parameters Changed	Parameters Dropped
BAR_WORKER_MAX		
BAR_WORKER_MAX		
BAR_BSALIB_PATH		
BAR_XPORT_COUNT		
BAR_XFER_BUFSIZE		
LOG_BACKUP_MODE		
LOG_BACKUP_MODE		
ISM_LOG_POOL		
BAR_ACT_LOG		
BAR_RETRY		
BAR_WORKER_MAX		

(2 of 2)

Multiple Coserver

For multiple coserver configuration, you can use the default values in the **onconfig.xps** template that is located in the **etc** directory in **INFORMIXDIR.**

Important: *Do not modify* ***onconfig.xps****. The database server provides the* ***onconfig.xps*** *configuration file as a template and not as a functional configuration.*

For more information on the template configuration files, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Figure 3-2
Multiple-Coserver Configuration Parameters

Parameters Added	Parameters Changed	Parameters Dropped
CONFIGSIZE	MIRRORPATH	DS_MAX_SCANS
COSERVER	ROOTNAME	
END	ROOTPATH	
PHYSSLICE	DBTEMPSPACE	
NODE	DS_TOTAL_MEMORY	
ROOTSLICE	MAX_PDQPRIORITY	
ISO_CURLOCKS	DBSERVERNAME	
DS_ADM_POLICY		
BAR_SM		
BAR_SM_NAME		
BAR_WORKER_COSVR		
BAR_DBS_COSVR		
BAR_LOG_COSVR		
BAR_ACT_LOG		
BAR_RETRY		
BAR_WORKER_MAX		
BAR_WORKER_MAX		
BAR_BSALIB_PATH		
BAR_XPORT_COUNT		
BAR_XFER_BUFSIZE		
LOG_BACKUP_MODE		
LOG_BACKUP_MODE		
ISM_LOG_POOL		

The sqlhosts File or Registry

Dynamic Server with AD and XP Options, Version 8.2, contains formatting changes and new options to the **sqlhosts** file or registry. The **dbservername** format and the options within the options field are changed from any prior version.

The dbservername Identifier

Within the **sqlhosts** file or registry in Dynamic Server with AD and XP Options, the **dbservername** identifies each connection coserver uniquely by the following format:

```
dbservername.coserver-number
```

Element	Description
dbservername	The value that you specify in the DBSERVERNAME or DBSERVERALIASES configuration parameter
coserver-number	The integer that you specify in each COSERVER configuration parameter

This form of the dbservername is referred to as a *coserver name*. Dynamic Server with AD and XP Options uses the DBSERVERNAME and coserver numbers specified in the ONCONFIG file to generate the following coserver names automatically:

```
xps.1
.
.
.
xps.8
```

New Options Syntax

The fifth field in the **sqlhosts** or registry fields, the **options** field, accommodates new options and future options.

The following list is a review of the **sqlhosts** or registry fields:

```
FIELD 1        FIELD 2     FIELD 3     FIELD 4       FIELD 5
DBSERVERNAME   NETTYPE     HOSTNAME    SERVICENAME   OPTIONS
```

The **dbservername** in field 1 is a key for connectivity information in the remaining fields in the **sqlhosts** file or registry.

The **options** field contains columns. Each column is separated by a comma or white space that represents an end of the column. Client and database server applications check each column to determine whether the option is supported in the database server release.

Tip: *If you maintain more than one version of the database server, use separate **sqlhosts** files or registry entries for older versions of the database server. Alternatively, you may use separate entries with an alias to the appropriate database server.*

For Dynamic Server with AD and XP Options, Version 8.2, an **sqlhosts** entry must exist for every coserver. The **sqlhosts** file or registry contains a line for each connection type that the database server provided, for each coserver that makes up the Dynamic Server with AD and XP Options, and for each Dynamic Server with AD and XP Options to which the client connects.

Figure 3-3 lists the Dynamic Server with AD and XP Options **sqlhosts** file or registry components.

Figure 3-3
Sqlhosts File or Registry Fields

dbservername	nettype	hostname	servicename	options
dbservername.coserver_1	onsoctcp	node1	sqlexec.1	e=x100,r=1,s=1
dbservername.coserver_2	onsoctcp	node2	sqlexec.2	k=1,r=1, b=200
⋮	⋮	⋮	⋮	⋮
dbservername.coserver_8	onsoctcp	node8	sqlexec.8	g=abcd,i=2300

In the sample **sqlhosts** file, the **options** field for the first coserver contains three options in three columns, as follows:

Column	Option
Column 1	`e=x100`
Column 2	`r=1`
Column 3	`s=1`

Important: *Informix recommends that you use field 5,* ***options****, for Dynamic Server with AD and XP Options, Version 8.2, for the following options only:* ***b****,* ***k****,* ***r****,* ***s****,* ***g****,* ***e****,* ***c****,* ***i****. Place any other options in subsequent columns. If you do not want any of these options, but do want other options, use* `k=1` *in column 5, which is the default. For example, for options for* ***dbservername.cosever_8****, use* `k=1, g=abcd,i=2300`.

For more information on the components of the **sqlhosts** file or registry, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Database Schema

The Dynamic Server with AD and XP Options, Version 8.2, database schema supports flexible fragmentation strategies, system-defined hash fragmentation, hybrid fragmentation, and unique serial columns across coservers. When you migrate to Dynamic Server with AD and XP Options, make sure to configure your database schema to take advantage of these features, which allow for the high degree of parallelism that Dynamic Server with AD and XP Options offers.

Hybrid Fragmentation

Hybrid fragmentation is a scheme that incorporates multiple fragmentation strategies. Prior to Dynamic Server with AD and XP Options, Version 8.2, table-fragmentation strategy is based on a single strategy, either hash, expression, or round-robin. Dynamic Server with AD and XP Options, Version 8.2, allows fragmentation by system-defined hash and you can combine two fragmentation strategies on the same table.

When you prepare to migrate to Dynamic Server with AD and XP Options, Version 8.2, define the dbspace or dbspaces where your fragmented tables reside, as the following example shows:

```
CREATE TABLE account
  (account_id integer,
   account_bal integer,
   account_date date,
   account_name char(30)
  )
 FRAGMENT BY HASH(account_date)
    IN account_dbsp1,
       account_dbsp2,
    .
    .
    .
account_dbspn;
```

This CREATE TABLE statement fragments the table across multiple dbspaces, which can reside on different coservers.You can create dbslices to manage the multiple dbspaces across coservers. The dbslices simplify administration of multiple dbspaces that reside on different coservers. For example, you can create a **dbslice account_dbslc** that includes **dbspaces account_dbsp1** to **account_dbspn**.

The following statement shows how the CREATE TABLE statement is simpler with the IN dbslice clause:

```
CREATE TABLE account
  (account_id integer,
   account_bal integer,
   account_date date,
   account_name char(30)
  ) FRAGMENT BY HASH(account_date)
    IN account_dbslc;
```

For more information on how to create a dbslice, refer to the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Hybrid fragmentation allows you to further fragment the table within each dbslice. You might want to fragment each time period by account number, as the following sample CREATE TABLE statement shows:

```
CREATE TABLE account
  (account_id integer,
   account_bal integer,
   account_date date,
   account_name char(30)
  )
 FRAGMENT BY HYBRID (HASH (account_date)) EXPRESSION
    account_id < 100 IN account_dbslc.1
    account_id >= 100 AND account_id > 200 IN account_dbslc.2
    .
    .
    .
    <expression_n> in account_dbslc.n;
```

For more information on hybrid fragmentation syntax, see the *Informix Guide to SQL: Syntax*. For more information and performance implications of using various fragmentation strategies, see the *Performance Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Unique Record Identifiers: Serial Columns

Serial columns in Advanced Decision Support and Extended Parallel Options are unique identifiers across coservers. Whereas in OnLine Dynamic Server or Informix Dynamic Server ROWIDs uniquely identify records, Advanced Decision Support and Extended Parallel Options do not use ROWIDs across multiple coservers as unique record identifiers. You can use ROWIDs in Advanced Decision Support and Extended Parallel Options, but they act as unique identifiers only if your tables are not fragmented.

Tip: *When you migrate to Dynamic Server with AD and XP Options, make sure your applications do not assume that ROWIDs are unique.*

Dynamic Server with AD and XP Options uses serial columns that are contiguous within each coserver to serve as unique identifiers, although a gap in numbers between coservers might occur. You can load values into serial columns, such as from the original data file, or use values that Dynamic Server with AD and XP Options generates automatically. However, when you create a table in Dynamic Server with AD and XP Options, you must specify that you want serial columns.

Design your Dynamic Server with AD and XP Options database schema to take advantage of fragmentation strategies and serial columns before you migrate to Dynamic Server with AD and XP Options. For general information about fragmentation and serial columns, see the *Informix Guide to Database Design and Implementation*.

Logging and Nonlogging Table Types

In Dynamic Server with AD and XP Options, no method exists to create a nonlogging database. The default logging mode is set to unbuffered logging. OnLine XPS, however, offers the ability to have logging and nonlogging *tables* and you can mix logging and nonlogging tables within the same database.

Dynamic Server with AD and XP Options has *permanent* and *temporary* tables with logging or nonlogging capabilities. Permanent tables have four classes: *raw*, *standard*, *operational*, and *static*. Temporary tables have two classes: *scratch* and *temp*. Both permanent and temporary tables can be altered from one class to another.

The logging mode of a table restricts the type of data load you can use for a table as well as the recoverability of the table.

Tip: *Use raw tables to initially load and scrub data and later alter the tables to another class, if necessary. Raw tables can be loaded in express mode only. For a description of express-load mode, see "Express Mode" on page 3-21.*

The following table lists the logging and loading characteristics of permanent and temporary table types. Express- and deluxe-loading modes are discussed in "Loading and Unloading Data" on page 3-20.

Table Type		Logging or Nonlogging	Load Mode Options
Permanent	Raw	Non logging	Express
	Standard	Logging	Deluxe
	Operational	Logging	Express or deluxe
	Static	Non logging	No loading
Temporary	Scratch	Non logging	
	Temp	Logging	

Permanent Tables

Use raw tables to initially load and scrub data. Raw tables use light appends that bypass the buffer cache and allow fast loading. They do not support indexes, referential constraints, rollback and recovery, or restoration from backup. You can use raw tables only with express-mode loads. For a description of express-load mode, see "Express Mode" on page 3-21.

Standard tables are similar to tables in a logged database in OnLine Dynamic Server or Informix Dynamic Server. They do not use light appends but support recovery and rollback. Standard tables allow restores. However, you cannot use express mode to load standard tables.

Use operational tables to load data from a source outside the database system. Operational tables use light appends—unbuffered, unlogged insert operations. They allow rollback and recovery, but do not allow restoration from backup. If indexes are enabled, deluxe-mode load is automatic. For descriptions of deluxe-load mode, see "Deluxe Mode" on page 3-21.

Use static tables for read-only operations. Static tables use light scans and do not need locking. They allow constraints and nonclustered indexes but do not allow rollback and recovery or restoration from backup. Static tables permit advanced indexing methods created especially for decision-support system (DSS) queries.

Static tables do not support data manipulation operations (inserts, updates, and deletes) and they cannot be used for loading. To load the tables, alter the table type to load and then change the table type back to static.

Temporary Tables

Scratch tables are unlogged and use light appends. They do not support indexes, referential constraints, or rollback.

Temporary tables are logged and support bulk operations, including light appends. Temporary tables also support indexes, referential constraints, and rollback but do not support recovery.

For information on correct SQL statement syntax and use when you create and change tables, see the *Informix Guide to SQL: Syntax*. For more details about table characteristics, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*. For more information on how to work with table types and fragmentation to design your database, see the *Informix Guide to Database Design and Implementation*.

Utilities

Dynamic Server with AD and XP Options uses new command-line utilities, **onutil** and **xctl**. The **onutil** utility consolidates much of the functionality of **oncheck** (**oncheck** is not supported in Dynamic Server with AD and XP Options), **onparams**, and **onspaces** into one command.

For a multiple coserver configuration, the **xctl** utility lets you execute command utilities and operating-system commands on one or more coservers. Informix recommends that you use **onutil** in Dynamic Server with AD and XP Options rather than other command utilities.

onutil

For a single coserver, use **onutil** to define or modify the following objects in Dynamic Server with AD and XP Options:

- Cogroups
- Dbslices
- Dbspaces
- Logical logs
- Logslices (an object name for a collection of logs)

Tip: *The **onutil** utility replaces the functionality of the **-cd**, **-pd**, **-pt**, **-pT**, **-pr, -pd, -cI, -ci, -pk, -pp, -pP,** and **-cc** options of **oncheck**.*

To define or modify physical logs, use **onparams**.

Figure 3-4 compares the **oncheck** commands to the **onutil** equivalent.

oncheck Command	onutil Command
-cd, -pd	CHECK TABLE DATA
-pt, -pT	CHECK TABLE INFO
-cr, -pr	CHECK RESERVED
-cI	CHECK INDEX WITH DATA
-ci, -pk	CHECK INDEX
-pp, -pP	DISPLAY PAGE
-cc	CHECK CATALOGS
-ce, -pe	CHECK SPACE

Figure 3-4
Comparable oncheck and onutil Commands

WIN NT

The installation procedure prepares a file, **setenv.cmd**, that sets the environment variables to their correct values. The **setenv.cmd** file is stored in the **%INFORMIXDIR%** directory. You must execute **setenv.cmd** before you can use any of the command-line utilities. If you choose **Command Line Utilities** from Informix Enterprise Command Center, the file executes automatically. You can also execute **setenv.cmd** from the command line.

xctl

For multiple coserver configuration, you can use **xctl** with **onstat**, **oninit**, and **onmode** to execute command utilities and operating-system commands on a per-coserver basis.

For example, use **xctl** to initiate Dynamic Server with AD and XP Options on all of your coservers:

```
xctl -C oninit -y
```

Several new options to **onstat** are available to check for OnLine XPS statistics. Use the following command to view the new **onstat** options:

```
xctl -onstat --
```

The following command brings down all coservers on a Dynamic Server with AD and XP Options instance:

```
xctl onmode -ky
```

For more information on **onutil** and **xctl**, refer to the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

WIN NT

Informix Enterprise Command Center

Informix Enterprise Command Center (IECC) for Dynamic Server with AD and XP Options runs on Windows NT. It allows the database server administrator to configure, control, and monitor the status of Dynamic Server with AD and XP Options. IECC provides a graphical interface that simplifies the process of database server administration and automates common database server administration functions.

IECC includes on-line help files. For additional information, see the *Informix Enterprise Command Center User Guide*. For information about how to install IECC, see the *Informix Enterprise Command Center Installation Guide.*

New Indexes

Dynamic Server with AD and XP Options, Version 8.2, supports *generalized-key* (GK*)* and *bitmap* indexes. GK indexes allow you to create indexes with key values that are:

- a subset of rows from a table.
- derived from an expression.
- a join of columns from multiple tables.
- a combination of various indexes on a table.

Bitmap indexes can store a list of record identifiers (ROWIDS) for key values in a compressed bitmap format. GK indexes can be bitmap indexes as well.

When you upgrade to Dynamic Server with AD and XP Options, Version 8.2, system catalog tables are added automatically to support GK indexes. However, since GK indexes are not supported in earlier versions of Dynamic Server with AD and XP Options, if you revert to an earlier version of Dynamic Server with AD and XP Options, drop GK indexes.

When you migrate to Dynamic Server with AD and XP Options, Version 8.2, bitmap indexes that you create with the USING BITMAP syntax are not accessible by older versions of the database server. If you revert to an earlier version of Dynamic Server with AD and XP Options, drop bitmap indexes.

For more information on GK indexes and bitmap indexes, refer to the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options* and the *Performance Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*. For strategic use of different index types, refer to the *Informix Guide to Database Design and Implementation*.

GLS

Global Language Support

Dynamic Server with AD and XP Options incorporates Global Language Support (GLS). GLS allows Dynamic Server with AD and XP Options to handle different languages, cultural conventions, and code sets for Asian, European, Latin American, and Middle Eastern countries. The *Informix Guide to GLS Functionality* provides a full description of GLS.

Loading and Unloading Data

Dynamic Server with AD and XP Options uses high-performance parallel loading with *external tables* to load and unload data. You can load tables from different sources and perform a variety of data-format conversions with external tables. The database server uses multiple threads to read data files in parallel and then convert the data into internal Informix format. A round-robin algorithm distributes the database server to the converter threads.

You can use the default delimiter (|) format to load and unload files with INSERT... SELECT statements. You can also use other formats. For loading instructions, see "Create a Database" on page 3-34.

The Dynamic Server with AD and XP Options high-performance parallel loading uses two modes: *express* and *deluxe*.

Express Mode

Express mode provides the highest performance during loads, because it uses light appends. Light-append operations append rows to a table and bypass the buffer cache that eliminates buffer-management overhead. The table is locked exclusively during an express load so that no other user can access the table during the load. You cannot use express mode on a table with BYTE and TEXT data.

Use express mode on tables that do not have active indexes or referential constraints. Express mode is the default mode.

Warning: *Data is not logged during an express-mode load so you cannot recover data automatically.*

Deluxe Mode

Deluxe mode combines fast parallel loading with an evaluation of indexes and unique constraints on a per-row basis. Deluxe mode is preferred if the cost of rebuilding an index is too high for the amount of data loaded. Data is logged during the load. If errors occur during a deluxe load, the load fails.

You can lock the table in shared mode to allow concurrent access by users, or alternatively, you can lock the table exclusively until the load is complete. Use deluxe mode if indexes, triggers, and referential constraints are enabled.

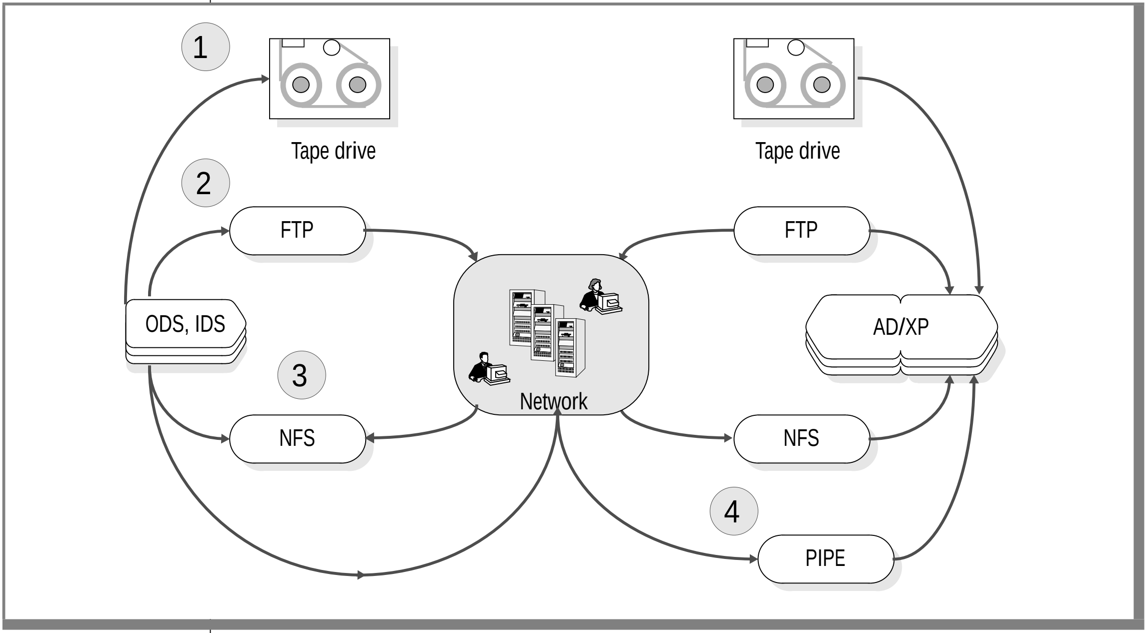
For complete documentation on external tables and deluxe- and express-mode loading syntax, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options* and the *Informix Guide to SQL: Syntax*.

Migrating from OnLine Dynamic Server 7.2 to Dynamic Server with AD and XP Options 8.2

You can use four options to migrate data from OnLine Dynamic Server to Dynamic Server with AD and XP Options, Version 8.2. Your data-migration options depend primarily on connectivity between the source and target database servers and on the amount of disk space available for staging files.

Figure 3-5 illustrates four options for migrating data from OnLine Dynamic Server or Informix Dynamic Server to Dynamic Server with AD and XP Options.

Figure 3-5
ODS to AD/XP Migration



- **Option 1.** To migrate from OnLine Dynamic Server when no connectivity is available between source and target database servers, unload tables to tape before you load to Dynamic Server with AD and XP Options.
- **Option 2.** To migrate when the required disk space for staging files is available on *both* the source database server *and* Dynamic Server with AD and XP Options, unload data from the source database to file.

  Use file transfer protocol (FTP) to transfer the file to Dynamic Server with AD and XP Options to load.
- **Option 3.** To migrate when you have sufficient disk space for staging files *only* on Dynamic Server with AD and XP Options or the source database, use network file system (NFS) mount points.

  For example, if Dynamic Server with AD and XP Options has enough disk space to handle your staging files, you can NFS mount a directory from Dynamic Server with AD and XP Options to OnLine Dynamic Server and unload your data from OnLine Dynamic Server to that NFS mounted directory.
- **Option 4.** You can use named pipes to support loading data to and unloading data from storage devices, including tape drives and direct network connections to mainframes.

  UNIX

  To convert data when you load, you can create a filter program that writes converted data to a named pipe. The database server then reads its input from the named pipe in one of the common formats. ♦

Tip: *Option 4 is the fastest. Options 2 and 3 are faster than option 1 because of the time required to read and write to tape.*

File-System Variations

File-system limitations vary between NFS and non-NFS file systems. You might need to break up large tables when you migrate to a new operating system.

For example, if you have a 3-gigabyte table, but your operating system only allows 2-gigabyte files, break up your table into separate files before you migrate. For more information, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Migration Procedures

This section describes the migration procedure from OnLine Dynamic Server, Version 7.2x, to Dynamic Server with AD and XP Options.

When you migrate to Dynamic Server with AD and XP Options, Version 8.2, complete the following steps, which are described in more detail in the sections that follow:

1. Configure and check available space.
2. Save copies of the current configuration files.
3. Close all transactions in the source database server.
4. Go to quiescent mode.
5. Verify the integrity of the data.
6. Verify the mode.
7. Make a final (level-0) backup.
8. Unload your data.
9. Bring the source database server off-line.

Important: *Repeat steps 1 through 9 for each instance of OnLine Dynamic Server, Version 7.2x, that you are migrating to Dynamic Server with AD and XP Options.*

10. Reconfigure the operating system, if necessary.
11. Install Dynamic Server with AD and XP Options, and copy **INFORMIXDIR** to each node.
12. Set environment variables on each node.
13. Update the ONCONFIG configuration files.
14. Update the **sqlhosts** or registry information.
15. Update the backup and restore configuration parameters.
16. Bring Dynamic Server with AD and XP Options on-line.
17. Use **onutil** to create cogroups and dbslices.
18. Create a database instance and load data.
19. Run UPDATE STATISTICS and build indexes.
20. Verify the integrity of the data.
21. Check rejection files for each table you load.

22. Recompile ESQL/C programs.
23. Make an initial backup.

Important: *Repeat steps 13 through 23 for each instance of Dynamic Server with AD and XP Options.*

Configure and Check Available Space

Configure your computer memory equally among all of the nodes. Since only one ONCONFIG file exists in Dynamic Server with AD and XP Options and the configuration parameter is configured globally, the amount of memory you configure in your ONCONFIG file is limited to the node with the smallest amount of memory. If your system memory is configured unequally, Dynamic Server with AD and XP Options cannot take advantage of the nodes that have extra memory.

When you calculate disk-space requirements for Dynamic Server with AD and XP Options, take into account the size of your tables. You need additional space for control information for your tables (approximately 60 bytes per page). Also consider disk space for the following needs:

- Root dbspace for all your nodes
- Physical logs
- Logical-log space.

  (If you require logging during loads, add the additional space.)
- Temporary dbspace
- Indexes
- Summary tables and index from your on-line transaction processing (OLAP) tools
- Mirroring
- Future growth
- File-system space for your operating system and Informix products
- Staging space for loading and unloading files
- Safe write area
- New system catalog tables
- New **sysmaster** tables

UNIX

Make sure you modify the kernel parameter on every node. ♦

For information on disk-space requirements for Dynamic Server with AD and XP Options, refer to the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*

Important: *If a dbspace has less free space available than it requires, either move a table from the dbspace to another dbspace or add a chunk to it. The dbspace estimates could be higher if you have an unusually large number of stored procedures or indexes in the database.*

Save Copies of the Current Configuration Files

Save copies of the configuration files, if they are present, for each instance of OnLine Dynamic Server. Keep the copies available for later use.

Figure 3-7 lists the configuration files for each operating system.

Figure 3-6
ODS Configuration Files

UNIX	Windows NT
$INFORMIXDIR/etc/$ONCONFIG	**%INFORMIXDIR%\etc\ONCONFIG**
$INFORMIXDIR/etc/onconfig	**%INFORMIXDIR%\etc\onconfig**
$INFORMIXDIR/etc/onconfig.std	**%INFORMIXDIR%\etc\onconfig.std**
$INFORMIXDIR/aaodir/adtcfg	**%INFORMIXDIR%\aaodir\adtcfg.***
$INFORMIXDIR/dbssodir/adtmasks	**%INFORMIXDIR%\dbssodir\adtmasks.***
$INFORMIXDIR/etc/sqlhosts	
$INFORMIXDIR/etc/tctermcap	
$INFORMIXDIR/etc/termcap	

UNIX

If you use ON-Archive to back up and restore your database server and logical logs, you must also copy and save the following files for possible reversion:

- **$INFORMIXDIR/etc/$ARC_CONFIG**
- **$INFORMIXDIR/etc/config.arc**
- **$INFORMIXDIR/etc/oper_deflt.arc** ♦

WIN NT

The Windows NT version of Dynamic Server with AD and XP Options does not use ON-Archive. Therefore, you do not need to copy these files. ♦

Close All Transactions in OnLine Dynamic Server

Communicate to client users how long you expect the database server to be off-line for the migration. Terminate all database server processes, and shut down the source database server.

To let users exit and shut down the system gracefully

1. Execute the **onmode –sy** command.
2. Wait for all users to exit.
3. Execute the **onmode –ky** command.

To perform an immediate shutdown

```
onmode -ky
```

Go to Quiescent Mode

Execute the following command to enter quiescent mode and initiate a fast recovery of your current database:

```
oninit -s
```

The **oninit –s** option rolls forward all committed transactions and rolls back all incomplete transactions since the last checkpoint and then leaves a new checkpoint record in the log with no open transactions pending. (For more information about fast recovery and quiescent mode, refer to your *Administrator's Guide*.)

You must execute **oninit -s** before you initialize your target database server. If the system is not left in a quiescent state, you receive the following error when you attempt to initialize Dynamic Server with AD and XP Options and it goes off-line:

```
Open transaction detected when changing log versions.
```

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of data before you make a level-0 backup. If you find any problems with the data, fix them before you make the backup. You can verify the integrity of the reserve pages, extents, system catalog tables, data, and indexes.

Obtaining a List of the Databases on Your Database Server

To obtain the database names, use the following statements with DB-Access:

```
DATABASE sysmaster;
SELECT name FROM sysdatabases;
```

Figure 3-7 lists the **oncheck** commands you can use to verify data integrity.

Figure 3-7
Commands for Verifying the Data Integrity

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *database_name*
Check data	`oncheck -cD` *database_name*
Check indexes	`oncheck -cI` *database_name*

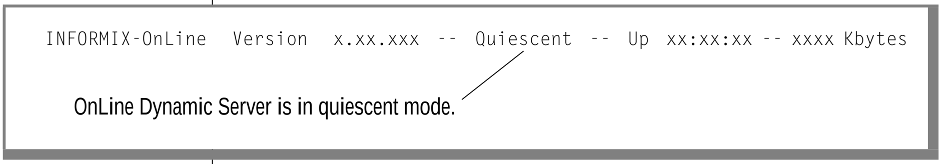
For information on **oncheck**, refer to your *Administrator's Guide*.

Verify the Mode

Before you make a backup, execute the following command to verify that your database server is in quiescent mode:

```
onstat -
```

The first line of the **onstat** output contains the status of the database server. Figure 3-8 shows that the database server is in quiescent mode.



```
INFORMIX-OnLine  Version  x.xx.xxx  --  Quiescent  --  Up  xx:xx:xx -- xxxx Kbytes
```

Figure 3-8
Example of onstat Status Line

Make a Final (Level-0) Backup

Use **ontape**, ON-Bar, or ON-Archive to make a level-0 backup and logical-log backup of OnLine Dynamic Server. If you use **ontape**, execute the following command to make a level-0 backup:

```
ontape -s
```

The tape parameters must specify a valid tape device. Be sure to retain and properly label the tape volume that contains the backup. (For more information about making backups, refer to the *INFORMIX-OnLine Dynamic Server Archive and Backup Guide*, Version 7.2.)

Once you have made a level-0 backup, perform a complete backup of the logical log, including the current logical-log file.

UNIX

If you use ON-Archive, execute the following command to make a full-system, level-0 backup:

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

WIN NT

Do not use ON-Archive before you migrate to Windows NT. Windows NT does not support ON-Archive. ♦

Unload Your Data

You can unload your data from your source database server into ASCII or internal Informix format, which yields the fastest unload. You can use the UNLOAD statement, the High-Performance Loader (HPL), or **dbexport**. The HPL is the fastest of these methods. For information about these options to unload your data, refer to Chapter 11, “Utilities for Data Migration.”

When unloading floats or small floats to a fixed-position file, if the internal precision of the float is greater than the fixed field can support, the number is truncated to fit the field. If you want more control over the formatted result of a float, use the SQL functions ROUND() or TRUNC() to define the output precision and to covert the float to a decimal.

Bring the Source Database Server Off-Line

Execute the following command to take OnLine Dynamic Server, Version 7.2x, to off-line mode:

```
onmode -ky
```

Bring your source database server off-line to ensure that all common files are inactive. Your source database server must be off-line because Dynamic Server with AD and XP Options uses the same files. You cannot install Dynamic Server with AD and XP Options if any of the files that it uses are active.

After you shut down your source database server, execute the following command to verify that it is off-line:

```
onstat -
```

Verify that you get the message `shared memory not initialized...` for off-line mode.

Important: *Make a final backup for each source database server instance that you plan to convert.*

UNIX

Reconfigure the Operating System

You might need to change some of the kernel parameters for your operating system before you install Dynamic Server with AD and XP Options. To reconfigure the operating system, follow the directions in the machine-notes file included on your Dynamic Server with AD and XP Options distribution media for your operating system.

For information on the location of the machine-notes file, refer to "Documentation Notes, Release Notes, Machine Notes" on page 20 of the Introduction.

Important: *Make sure you modify the kernel parameter on every node.*

Install Dynamic Server with AD and XP Options

To install Dynamic Server with AD and XP Options on UNIX, you must log in as user **root** or **informix.** To install Dynamic Server with AD and XP Options on Windows NT, you must be a member of the **Informix-Admin** group. Set the **INFORMIXDIR** environment variable to the directory where you plan to install Dynamic Server with AD and XP Options. Install Dynamic Server with AD and XP Options on the node that contains the connection coserver in the directory that was NFS mounted on all the other nodes. Install the entire distribution on a single node within a file system that is shared across all the nodes that are assigned to Dynamic Server with AD and XP Options.

Check that the file system can hold the entire Dynamic Server with AD and XP Options distribution of approximately 180 megabytes. Export this file system with write access as user **root** or **informix** over the NFS, and mount it to the same mount point on every node.

Warning: *If you install Dynamic Server with AD and XP Options in the same directory where OnLine Dynamic Server resided, the installation script overwrites the older files. If you wish to preserve your OnLine Dynamic Server files, you must install Dynamic Server with AD and XP Options in a different directory on UNIX or on different computers on Windows NT.*

Check the release notes for information about the proper operating-system patches, recommended shared-memory parameters, and configurations that are required for successful installation and operation of the database server.

UNIX

The release notes are in the **$INFORMIXDIR/release/en_us/0333** directory. ♦

WIN NT

The release notes are in **%INFORMIXDIR%\release\en_us\0333**. ♦

Make any required changes that the release notes recommend on every node. For complete information on how to install Dynamic Server with AD and XP Options, refer to the *Performance Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Set Environment Variables

After you install Dynamic Server with AD and XP Options and before you invoke DB-Access, set up the following environment variables on every node:

- **INFORMIXDIR**
- **INFORMIXSERVER**
- **ONCONFIG**
- **PATH**
- **INFORMIXSQLHOSTS**

  UNIX

  The client application looks for the **sqlhosts** file in the **$INFORMIXDIR/etc** directory. However, you can use the **INFORMIXSQLHOSTS** environment variable to change the location or name of the **sqlhosts** file. ♦

  WIN NT

  The client application looks for connectivity information in a key in the Windows NT registry called HKEY_LOCAL_MACHINE\SOFTWARE\Informix\SQLHOSTS. ♦

Set the environment variable **PATH** so that the local directory that contains the Informix commands is searched before the **INFORMIXDIR** directory.

The installation script installs Dynamic Server with AD and XP Options into the **INFORMIXDIR** directory specified for user **root** or **informix** on UNIX platforms or for a member of the **Informix-Admin** group on Windows NT platforms. The installation script does not bring Dynamic Server with AD and XP Options on-line.

Copy Utilities

To ensure rapid and proper initialization, create local copies of the following utilities on each node:

- **oninit**
- **onmode**
- **onstat**

When you finish copying the utilities to each node, you can log out as **root**.

Update the ONCONFIG Configuration Files

You can customize your ONCONFIG configuration file and environment variables to take advantage of the new features introduced by Dynamic Server with AD and XP Options. After you observe the performance of Dynamic Server with AD and XP Options, you might want to make further adjustments.

For information on how to configure Dynamic Server with AD and XP Options, refer to the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*. For information about environment variables, refer to the *Informix Guide to SQL: Reference*.

Bring Dynamic Server with AD and XP Options On-Line

Execute the following command to bring Dynamic Server with AD and XP Options, Version 8.2, on-line for the first time:

```
xctl -C oninit -iy
```

Review the message logs to verify that your **sysmaster** database and **sysmaster** catalog tables were created successfully.

Check your Dynamic Server with AD and XP Options message log to verify that all coservers are up. To view the message log, run:

```
onstat -m
```

To verify that all your coservers are up and running, execute:

```
xctl onstat -
```

For information about messages in the message log, refer to the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Important: *If the message file indicates problems, solve the problems before you continue to the next step.*

Use onutil to Create Cogroups and Dbslices

Create cogroups and dbslices of equal sizes across all coservers. When you create a dbslice, you specify the cogroup name so that Dynamic Server with AD and XP Options knows the coservers on which to create dbspaces. For example, you might create a dbslice from an accounting cogroup. The following example shows how to create a cogroup and a dbslice:

```
% onutil
1> CREATE COGROUP acctg_group
.
.
.
5> CREATE DBSLICE acctg_dbslc
6>      FROM acctg_group ...
```

You do not need to specify the names explicitly for all of the individual dbspaces that are associated with the partitioned tables. Dynamic Server with AD and XP Options generates the dbspace names for you.

Run **xctl** to verify that all your dbspaces were created correctly:

```
xctl onstat -d
```

Create a Database

Create a database instance on Dynamic Server with AD and XP Options and choose the optimal loading scheme from the data-loading options discussed in "Loading and Unloading Data" on page 3-20. After you install Dynamic Server with AD and XP Options, you can create a database using DB-Access as your client application or issue SQL statements to connect to and disconnect from an Dynamic Server with AD and XP Options database server.

You can start IECC from a Windows NT console to customize one or more Dynamic Server with AD and XP Options database servers. For instructions for how to customize a database server, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Load Data from External Tables

The following section describes how to load data into Dynamic Server with AD and XP Options external tables and convert it into internal Informix format. You can issue these SQL statements in the DB-Access utility or embed them in INFORMIX-ESQL/C.

Use a CREATE TABLE statement to create a table in your Dynamic Server with AD and XP Options database to which you plan to load the data from OnLine Dynamic Server. For example:

```
CREATE TABLE account
    (account_id integer,
    account_bal integer,
    account_date date,
    account_name char(30)
    )
 FRAGMENT BY HASH(account_date)
    IN account_dbsp1,
       account_dbsp2,
    .
    .
    .
    account_dbspn;
```

Issue a series of SQL statements to create external tables for loading data. The CREATE EXTERNAL TABLE statement describes the location of the external file (on file or from pipe) and the format of external data. When you load with external tables, you can select FILE or PIPE for named pipes for your data source. You can load DELIMITED format data, FIXED format data or INFORMIX internal format data (which is created when your unload data in this format). Loading with INFORMIX internal format is fast since no conversion is required. Delimited or fixed-format data can be in ASCII or EBCDIC code sets.

The external table statement also can describe data-conversion information and specify a file for error and diagnostic information.

For example:

```
Create EXTERNAL TABLE extTableName
    SAME AS sourceTableName
    USING
    FORMAT “informix”,
    DATAFILES
(
    “DISK: COGROUP_ALL:/data/file.%c”,
    REJECTFILE “/pload/reject”
);
```

If the files you are loading were unloaded from a database in Informix internal format, specify an Informix format in the USING clause. The SAME AS clause creates the external table with the same definitions for the columns of the data files from the existing database.

Specify DATAFILES

The USING DATAFILES clause specifies the file type, coserver or cogroup, and the location of the file, along with the file format definition and location for a reject file for error and diagnostic information. The USING DATAFILES clause takes double quoted strings separated by commas. The DATAFILES statement has three parts, separated by colons: the data type, the data location, and the absolute path to the file.

To move data between external tables and internal tables, issue data manipulation language (DML) SQL statements. Load data files into the Dynamic Server with AD and XP Options database by issuing the INSERT AND SELECT statements. The INSERT and SELECT statements map the movement of the external data from or to the database table. The data in the data files is loaded from the external table into an internal table using the INSERT into SELECT statements. For example:

```
LOCK TABLE IN EXCLUSIVE MODE; # for deluxe mode

INSERT INTO account
(account_id, account_bal, account_date date, account_name)
SELECT *
FROM extTableName
USING
(FORMAT “informix”,
DATAFILES
(
“DISK:3:/tmp/data.1”
“DISK:5:/tmp/data.1”
)
WHERE account_name [1] = “A”;
```

Tip: *Specify the format* `informix` *only for data files you unload from an Informix database. The WHERE clause can specify which data file rows to load.*

Set Log File and Maximum Errors

You can set a log file that stores session information and statistics gathered about each load and unload when it completes. The log file also lists reject files. For information on reject files, see “Check Rejection Files” on page 3-39. If the WITH APPEND keywords are used, the new log information is appended to an existing file; otherwise the file is truncated. To set this log file, issue the following statement:

```
SET PLOAD FILE TO filename
[WITH APPEND];
```

Tip: *You can only have one file open at a time. If you issue multiple SET PLOAD FILE statements, the last one you issue is in effect.*

To control the maximum number of errors allowed per coserver before a load is aborted, you can set the MAXERRORS option in the external table. Include the following statement in your CREATE EXTERNAL TABLE statement:

```
MAXERRORS maxNumErrPerCoserver
```

For more information on the CREATE EXTERNAL TABLE statement, refer to the *Informix Guide to SQL: Syntax*.

You can also use **onstat -g xmp** and **onstat -g dfm** to monitor the query segments and data flow. For more information, see the *Performance Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Run UPDATE STATISTICS and Build Indexes

Run UPDATE STATISTICS on all of the tables you load into your Dynamic Server with AD and XP Options database.

Build indexes on the tables in Dynamic Server with AD and XP Options that were indexed in OnLine Dynamic Server. You might want to wait before you build indexes to see if your queries run fast enough without them.

Verify the Integrity of the Data

After Dynamic Server with AD and XP Options finishes converting the system catalog tables, use **onutil** to verify that no data was corrupted in the migration process. The **onutil** commands are not SQL statements; neither DB-Access nor any other SQL utility or application supports them. Only user **informix** or **root** can use the CHECK and DISPLAY DATA clauses. The **onutil** CHECK options place a shared lock on tables when they check indexes. They also place shared locks on system catalog tables when they check them. You can verify the integrity of the reserve pages, extents system catalog tables, data, and indexes, as shown in Figure 3-9.

Figure 3-9
Commands for Verifying Data Integrity

Action	onutil Command
Check reserve pages	`onutil CHECK RESERVED DISPLAY DATA`
Check extents	`onutil CHECK SPACE DISPLAY DATA`
Check system catalog tables	`onutil CHECK CATALOGS`
Check data	`onutil CHECK TABLE DATA DISPLAY DATA`
Check indexes	`onutil CHECK INDEX DISPLAY DATA`

You might want to test and run your queries on Dynamic Server with AD and XP Options and make sure they produce the same results.

For more information on **onutil**, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Make an Initial Backup

Use your Dynamic Server with AD and XP Options backup and restore tool (ON-Bar) to make a level-0 backup. Do not overwrite the tapes you used earlier when you made your final backup of your source database server.

Dynamic Server with AD and XP Options supports unloading to pipes, so you can unload directly to a tape device. You do not need to unload data first to disk.

You can use external tables to perform a backup. This process requires adequate disk space to stage your data for loading and unloading. To back up, unload a table to an external table and back up to tape. To restore, load data from your external tables.

For more information about ON-Bar, see the *Backup and Restore Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Important: *Do not restore the backed up logical-log files from OnLine Dynamic Server for Dynamic Server with AD and XP Options.*

Check Rejection Files

The REJECTFILE file specifies which rows were rejected during a load for each coserver. Each coserver must have a unique REJECTFILE name. Enter the REJECTFILE parameter in the USING clause of the EXTERNAL table definition as shown in the example on page 3-35. For example:

```
REJECTFILE absolutePathName
```

Warning: *Do not use shared files across coservers to ensure that the data in the reject files retain integrity.*

Check the REJECTFILE file from your load command and see if any records were not loaded during the load. If necessary, manually update tables. You can also monitor message files using **onstat -g xmf**. For more information on how to monitor options, see the *Performance Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Recompile ESQL/C Programs

INFORMIX-ESQL/C allows programmers to embed SQL statements directly into a C program. Once your database server migration is complete, recompile your ESQL/C programs. Make sure to set the environment variables before you invoke an ESQL/C program.

When the ESQL/C program is recompiled, check the functionality of your ESQL/C programs to make sure they run as you expect them to. For more information, see the *INFORMIX-ESQL/C Programmer's Manual*.

Migration Complete

When you finish the level-0 backup, the migration process is complete and users can use Dynamic Server with AD and XP Options to access data safely.

Once you successfully migrate to Dynamic Server with AD and XP Options, you might want to seek ways to obtain maximum performance. If you created sample queries for comparison, you can use them to characterize the performance differences between Informix Dynamic Server or OnLine Dynamic Server, Version 7.2, and Dynamic Server with AD and XP Options. The results of these comparisons might suggest adjustments to configuration parameters, or to the layout of databases, tables, and chunks. For details on performance topics, refer to the *Performance Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Reverting AD/XP 8.2 to OnLine Dynamic Server 7.2

Once you migrate from OnLine Dynamic Server, Version 7.2, to Dynamic Server with AD and XP Options, Version 8.2, you cannot revert back automatically. To revert, you unload the data from Dynamic Server with AD and XP Options, Version 8.2, into an external table and then load it back into OnLine Dynamic Server.

Unloading to External Tables

To unload data to an external file, issue SELECT and INTO EXTERNAL statements. You can implicitly or explicitly specify an external table. (If you are loading data, an explicit definition is mandatory.) The SELECT and INTO EXTERNAL statements create a default external table description for unloading data. Issue the following SQL statement:

```
SELECT *
FROM customer
WHERE customerNum > 100
AND lastName [1] = “A”
INTO EXTERNAL TABLE custExtII
USING
(
FORMAT “informix”,
DATAFILES
( “DISK:1:/tmp/dat.out”,
“DISK:2:/tmp/dat.out”,
“DSIK:3:/tmp/dat.out”
));
```

In this example, the external table is implicitly defined when the table is unloaded. If the external table already exists, you can use the following syntax:

```
INSERT INTO extTableName
SELECT *
FROM tableName
WHERE...;
```

Once the data is unloaded into external tables, you can use OnLine Dynamic Server load utilities to load the data. For load utility information, see Chapter 11, “Utilities for Data Migration.”

Migrating Between OnLine XPS 8.1x and AD/XP 8.2

The migration from OnLine XPS, Version 8.11, to Dynamic Server with AD and XP Options, Version 8.2, is automatic and transparent. Dynamic Server with AD and XP Options, Version 8.2, incorporates some new internal formats, but the database server updates the tables for you.

Once you migrate from OnLine XPS, Version 8.11, to Dynamic Server with AD and XP Options, Version 8.2, you cannot revert back automatically. To revert, unload the data from Dynamic Server with AD and XP Options, Version 8.2, into an external table and then load it back into OnLine XPS, Version 8.11. You can use a **dbxpsldr.ec** utility to perform much of the migration for you.

Migration Procedures

This section describes the migration procedure between OnLine XPS, Version 8.1x, and Dynamic Server with AD and XP Options, Version 8.2.

1. You can run the **dbxpsldr.ec** utility, located in **%INFORMIXDIR/demo/xmp** on UNIX and in **%INFORMIXDIR%\demo\xmp** on Windows NT, on the source database to create three output SQL scripts:
   - **x_*dbname*.sql** creates external table definitions for all internal user tables in the database *dbname*.
   - **u_*dbname*.sql** creates an unload script that unloads all the tables.

- **l_*dbname*.sql** creates a load script. The load script alters tables to *raw* or *operational* and drops associated indexes for each table. It loads each table from the external data and alters each table back to the original type, except where not possible, and re-creates all associated indexes. Tables that have primary or unique keys are altered to the operational mode instead of raw. If the table type is standard, the load script reports this in a comment, but changes the mode to operational since you must run a level TIP archive before you promote a table type to standard.

  You can use ESQL/C to compile the **dbxpsldr.ec** utility:

  ```
  esql -o dbxpsldr dbxpsldr.ec
  ```

The **dbxpsldr** utility has the following options:

```
dbxpsldr dbname [-a] [-x exttab_file]
[-l load_file]
[-u unload_file]
```

Option	Description
-a	Use this option to designate data in ASCII-delimited format. The default is Informix format.
-x *filename*	Use this option to name the external table script. The default is **x_*dbname*.sql.**
-l *filename*	Use this option to name the load script. The default is **l_*dbname*.sql.**
-u *filename*	Use this option to name the unload script. The default is **u_*dbname*.sql.**

2. Create external tables with the same schema from the source database. For example:

```
Create external table x_tab
sameas tab
using (format "informix",
datafiles ("disk:cogroup_all:$DIR_DBNAME/
    tab.tbl.%c"));
```

   Edit this file to indicate the location for the data files for this database. Make sure the data files point to the location of the unloaded data. For example, if you are using a pipe on coserver 1 and 2, use:

```
datafiles
("pipe:1:/data/order.tab.1", "pipe:2:/data/
    order.tab.2"));
```

   The default for the template is to use **cogroup_all** and to use the filename expansion macro **%c** to append the coserver number for each file to make each data file unique on the disk. If necessary, modify the data file locations in the **x_*dbname*.sql** script and run the **x_*dbname*.sql** and **u_*dbname*.sql** scripts.

3. Run **dbschema**, as follows:

```
dbschema -ss -d databasename outfilename
```

   The **-ss** option saves all table fragmentation and permissions information.

4. Archive the **dbschema** files, scripts, and the unloaded data.

5. Before you load your data, build the dbspaces and dbslices for the new database, and create the tables.

6. Shut down your source database server and install and initialize the target database server.

7. Run the scripts that **dbschema** created to make the tables.

8. On the target database server, run the **x_*dbname*.sql** and the **l_*dbname*.sql** scripts.

9. Perform a level-0 archive.

10. Alter table types to standard if desired.

11. Run UPDATE STATISTICS as needed.

Migrating from Previous 8.x Versions

If you are upgrading to Dynamic Server with AD and XP Options, Version 8.2, from any version of 8.x earlier than 8.11.UF1, and you did not restart the database server with **oninit -iy**, drop and re-create the **sysutils** database. A new column has been added to a table in the **sysutils** database.

UNIX

Log in as user **informix** or **root**, and execute **$INFORMIXDIR/etc/bldutil.sh**. The **bldutil.sh** command destroys all backup history. ♦

If you want to retain the backup history, unload and re-create the **sysutils** tables. If you are moving from 8.11.UF1 or later, you do not need to re-create the **sysutils** tables.

For more information about the **sysutils** database, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*.

Chapter

4

Migrating the 7.2 Database Server to Version 9.1x

This chapter describes the procedure to migrate between OnLine Dynamic Server, Version 7.2x, and INFORMIX-Universal Server or Informix Dynamic Server with Universal Data Option.

This chapter covers the following topics:

- Upgrading OnLine Dynamic Server, Version 7.2x, to Universal Server or Informix Dynamic Server with Universal Data Option
- Reverting Universal Server or Informix Dynamic Server with Universal Data Option to OnLine Dynamic Server, Version 7.2

Important: *To migrate to INFORMIX-Universal Server or Informix Dynamic Server with Universal Data Option from OnLine Dynamic Server or Informix Dynamic Server, you must first migrate to OnLine Dynamic Server, Version 7.2. Follow the instructions in this chapter to migrate to Universal Server or Informix Dynamic Server with Universal Data Option.*

You can upgrade automatically from OnLine Dynamic Server, Version 7.10.UD1, through Version 7.2x and Informix Dynamic Server, Version 7.3, to Universal Server or Informix Dynamic Server with Universal Data Option and revert back.

Preparing to Migrate to Universal Server or Informix Dynamic Server with Universal Data Option

Informix suggests that you observe the following precautions when you migrate to Universal Server or Informix Dynamic Server with Universal Data Option:

- Check the release notes for information about the proper operating-system release and any patches that you need for successful installation and operation of the database server.

  The release notes are in one of the following directories:

  UNIX

  - **$INFORMIXDIR/release/en_us/0333**. ♦

  WIN NT

  - **%INFORMIXDIR%\release\en_us\0333**

    Release Notes appear in the Informix folder. To display this folder, choose **Start→Programs→Informix** from the Task Bar. ♦

  In UNIX, retain both versions of the Informix product software on disk (if you have enough disk resources). On Windows NT, on versions prior to Version 7.3, you cannot retain two versions of the Informix product on disk.
- Retain the installation tapes for UNIX or the CDs for Windows NT from both versions of the Informix product software.
- Perform a level-0 backup of all dbspaces and blobspaces with OnLine Dynamic Server. After you complete the migration, perform another level-0 backup with Universal Server.
- Use a test instance of your database server to test the installation and migration procedures.

For additional information, refer to both your *Installation Guide* and *Getting Started with INFORMIX-Universal Server*.

Changes Introduced by Universal Server

This section describes the changes introduced by Universal Server that affect either migration or initial configuration.

Environment Variables

Universal Server introduces the following new environment variables that were not available in OnLine Dynamic Server, Version 7.2:

- INFORMIXCONCSMCFG
- INFORMIXKEYTAB

Review the descriptions of these environment variables for communications support services (CSM) to determine whether you need to set them. The *Informix Guide to SQL: Reference* manual describes these environment variables.

Configuration Parameters

Universal Server introduces new configuration parameters that might affect your installation. You might also need to adjust the values of existing parameters. The following new configuration parameters for Universal Server are described in the *INFORMIX-Universal Server Administrator's Guide*:

- HETERO_COMMIT
- SBSPACENAME
- VPCLASS

The SBSPACENAME parameter specifies a default *sbspace* for the storage of *smart large objects*. For more information on smart large objects and sbspaces, refer to "New Database Storage Spaces" and to the *INFORMIX-Universal Server Administrator's Guide*.

The VPCLASS parameter combines options from several virtual-processor configuration parameters and enables you to configure a virtual processor class with a single parameter. Consequently, if you use the VPCLASS parameter, be aware that it affects the following parameters, which you may need to change or delete:

- AFF_NPROCS
- AFF_SPROC
- NOAGE
- MULTIPROCESSOR
- NUMAIOVPS
- NUMCPUVPS
- SINGLE_CPU_VP

New Database Storage Spaces

Universal Server introduces the following two new storage spaces:

- Sbspaces
- External spaces

An sbspace provides storage space for smart large objects. An external space is a space outside of the direct control of Universal Server, and one which contains data that you want to include in a database. If you use external spaces, you must also define the access methods that Universal Server will use to retrieve the external data. For more information on external spaces, refer to the *INFORMIX-Universal Server Administrator's Guide*. For more information on user-defined access methods, refer to the *Virtual-Table Interface Programmer's Manual*.

New Options Syntax in the sqlhosts File or Registry

Informix Dynamic Server with Universal Data Option, Version 9.13, introduces formatting changes and new options to the **sqlhosts** file or registry.

The fifth field in the **sqlhosts** or registry fields, the **options** field, was modified to accommodate new options and future options.

The following table is a review of the **sqlhosts** or registry fields.

FIELD 1	FIELD 2	FIELD 3	FIELD 4	FIELD 5
dbservername	nettype	hostname	servicename	options

The **dbservername** in field 1 is a key for connectivity information in the remaining fields in the **sqlhosts** file or registry.

The **options** field contains columns. Each column is separated by a comma or white space that represents an end of the column. Client and database server applications check each column to determine whether the option is supported in the database server release.

For Informix Dynamic Server with Universal Data Option, Version 9.13, or higher, the **sqlhosts** file contains a line for each connection type that the database server provides and for each instance of Dynamic Server with UD Option to which the client connects.

Tip: *If you maintain more than one version of the database server, use separate* ***sqlhosts*** *files or registry entries for older versions of the database server. Alternatively, you can use separate entries with an alias to the appropriate database server.*

The **sqlhosts** file or registry contains a line for each connection type that the database server provides.

Figure 4-1 lists the Informix Dynamic Server with Universal Data Option Version 9.13 **sqlhosts** file or registry components.

Figure 4-1
sqlhosts File or Registry Fields

dbservername	nettype	hostname	servicename	options
dbservername.server1	olsoctcp	node1	15000	k=1 csm=(DCECSM),abc=def

In the sample **sqlhosts** file, the **options** field contains three options contained in three columns:

Column	Option
Column 1	`k=1`
Column 2	`csm=(DCECSM)`
Column 3	`abc=def`

Important: *Informix recommends that you use field 5,* ***options****, for Informix Dynamic Server with Universal Data Option,Version 9.13, or higher, for the following options only:* ***b****,* ***k****,* ***r****,* ***s****. If you do not want any of these options but do want other options, use* `k=1` *in column 5, which is the default. Informix Dynamic Server with Universal Data Option, Version 9.13, or higher allows key values with more than one character. Place other options in subsequent columns.*

If you use multiple versions of the database server, you must maintain two separate **sqlhosts** files or registries. For more information on the components of the **sqlhosts** file or registry or how to define two **sqlhosts** files or registries, see the *INFORMIX-Universal Server Administrator's Guide.*

GLS

Global Language Support

Informix Dynamic Server with Universal Data Option and INFORMIX-Universal Server incorporates Global Language Support (GLS). GLS lets Universal Server handle different languages, cultural conventions, and code sets for Asian, European, Latin American, and Middle Eastern countries. The *Informix Guide to GLS Functionality* provides a full description of GLS.

Migrating from OnLine Dynamic Server 7.2x to the 9.1x Database Server

This section describes the migration procedure from OnLine Dynamic Server, Version 7.2x, to Universal Server or Dynamic Server with UD Option, on UNIX.

Migration Procedures

When you migrate to Universal Server or Dynamic Server with UD Option, you can install and test a database server instance with the same configuration files, environment variables, and **sqlhosts** information that you used for your source database server. After you install Universal Server or Dynamic Server with UD Option and verify that it works, you might want to modify configuration files and environment variables to take advantage of the Universal Server features. For more information, refer to *Getting Started with INFORMIX-Universal Server* and the *INFORMIX-Universal Server Administrator's Guide*.

When you migrate to Universal Server or Dynamic Server with UD Option, complete the following steps, which are described in more detail in the sections that follow:

1. Check available space.
2. Save copies of the current configuration files.
3. Close all transactions with the source database server.
4. Initiate a fast recovery.
5. Verify the integrity of the data.
6. Verify the mode.
7. Make a final (level-0) backup.
8. Bring the current OnLine Dynamic Server off-line.

Important: *Repeat steps 2 through 8 for each instance of OnLine Dynamic Server Version 7.2, that you are migrating to Universal Server or Dynamic Server with UD Option.*

9. Reconfigure the operating system, if necessary.
10. Install Universal Server or Dynamic Server with UD Option.
11. Verify that environment variables are set correctly.
12. Update the ONCONFIG configuration files.
13. Update the backup and restore configuration parameters.
14. Bring the target database server on-line.
15. Verify the integrity of the data.
16. Make an initial (level-0) backup under your target database server.

Important: *Repeat steps 12 through 16 for each instance of Universal Server or Dynamic Server with UD Option that you run on the computer.*

17. Install and configure any DataBlade modules that you are adding to Universal Server or Dynamic Server with UD Option.

Check Available Space

UNIX

Universal Server or Dynamic Server with UD Option requires 3000 free pages of logical-log space (approximately 6000 kilobytes for a 2-kilobyte page size) to build the **sysmaster** database. ♦

WIN NT

Universal Server or Dynamic Server with UD Option requires 1500 to 3000 free pages of logical-log space (approximately 6000 kilobytes for a 4-kilobyte page size) to build the **sysmaster** database on Windows NT. ♦

Universal Server or Dynamic Server with UD Option requires approximately 2000 kilobytes more space per database than OnLine Dynamic Server. The extra space is used for new system catalog tables and built-in functions that support the extensibility features of Universal Server or Dynamic Server with UD Option.

When you initialize Universal Server or Dynamic Server with UD Option on an existing OnLine Dynamic Server or Informix Dynamic Server root dbspace, it automatically upgrades the **sysmaster** database. Each database is individually upgraded when it is first accessed. To upgrade each database successfully, you must ensure that the 2000 kilobytes per database is available in each dbspace.

1. Calculate the amount of free space that each dbspace requires, where *n* is the number of databases in the dbspace and *X* is the amount of free space they require:

   ```
   X kilobytes free space = 2000 kilobytes * n
   ```

2. Check the amount of free space in each dbspace to determine whether you need to add more space.

Use the following SQL statements to determine the free space required and the free space available. These statements return the free-space calculation in page-size units. The variable **free_space_req** displays the free space required and the variable **free_space_avail** displays the free space available.

The following SQL statement shows how to determine the free space that each dbspace requires:

```
DATABASE sysmaster;
SELECT partdbsnum(partnum) dbspace_num,
       trunc(count(*) * 2000) free_space_req
    FROM sysdatabases
GROUP BY 1
ORDER BY 1;
```

The following SQL statement queries the **syschunks** table and displays the free space available for each dbspace:

```
SELECT dbsnum dbspace_num, sum(nfree) free_space_avail
    FROM syschunks
GROUP BY 1
ORDER BY 1;
```

Important: *If less free space is available than the dbspace requires, either move a table from the dbspace to another dbspace, or add a chunk to it.*

The dbspace estimates could be higher if you have an unusually large number of stored procedures or indexes in the database.

Save Copies of the Current Configuration Files

Save copies of the configuration files for each instance of your source database server. Keep the copies available to use later. Save the configuration files that Figure 4-2 lists, if they exist.

Figure 4-2
ODS or IDS Configuration Files

UNIX	Windows NT
$INFORMIXDIR/etc/$ONCONFIG	**%INFORMIXDIR%\etc\ONCONFIG**
$INFORMIXDIR/etc/ONCONFIG.STD	**%INFORMIXDIR%\etc\ONCONFIG.STD**
$INFORMIXDIR/aaodir/adtcfg	**%INFORMIXDIR%\aaodir\adtcfg.***
$INFORMIXDIR/dbssodir/adtmasks	**%INFORMIXDIR%\dbssodir\adtmasks.***
$INFORMIXDIR/etc/SQLHOSTS	
$INFORMIXDIR/etc/TCTERMCAP	
$INFORMIXDIR/etc/TERMCAP	

UNIX

If you use ON-Archive to back up and restore your source database server and the logical logs, you must also copy and save the configuration files in the following list:

- **$INFORMIXDIR/etc/$ARC_CONFIG**
- **$INFORMIXDIR/etc/config.arc**
- **$INFORMIXDIR/etc/oper_deflt.arc** ♦

WIN NT

The Windows NT version of Universal Server or Dynamic Server with UD Option does not use ON-Archive. Therefore, you do not need to copy these files. ♦

Close All Transactions

Communicate to client users how long you expect the database server to be off-line for the migration. Terminate all database server processes, and shut down your source database server.

To let users exit and shut down the system gracefully

1. Execute the **onmode –sy** command.
2. Wait for all users to exit.
3. Execute the **onmode –ky** command.

To perform an immediate shutdown

```
onmode -ky
```

Initiate a Fast Recovery

Execute the following command to enter quiescent mode and initiate a fast recovery of your current database:

```
oninit -s
```

The **oninit –s** option rolls forward all committed transactions and rolls back all incomplete transactions since the last checkpoint and then leaves a new checkpoint record in the log with no open transactions pending. (For more information about fast recovery, refer to your *Administrator's Guide*.)

You must execute **oninit -s** before you initialize Universal Server or Dynamic Server with UD Option. If the system is not left in a quiescent state, you receive the following error when you attempt to initialize the database server and it goes off-line:

```
Open transaction detected when changing log versions.
```

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of data before you make a level-0 backup. If you find any problems with the data, fix them before you make the backup. You can verify the integrity of the reserve pages, extents, system catalog tables, data, and indexes.

Obtaining a List of the Databases on Your Database Server

To obtain the database names, use the following statements with DB-Access:

```
DATABASE sysmaster;
SELECT name FROM sysdatabases;
```

Figure 4-3 lists the commands that verify data integrity.

Figure 4-3
Commands for Verifying the Data Integrity

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *database_name*
Check data	`oncheck -cD` *database_name*
Check indexes	`oncheck -cI` *database_name*

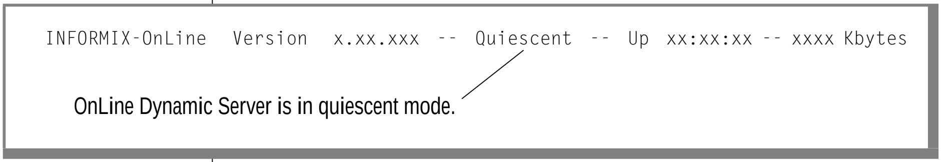
For information on **oncheck**, refer to your *Administrator's Guide*.

Verify the Mode

Before you make a backup, execute the following command to verify that your source database server is in quiescent mode:

```
onstat -
```

The first line of the **onstat** output contains the status of your source database server. Figure 4-4 shows that the database server is in quiescent mode.



```
INFORMIX-OnLine  Version  x.xx.xxx  --  Quiescent  --  Up  xx:xx:xx -- xxxx Kbytes
```

OnLine Dynamic Server is in quiescent mode.

Figure 4-4
Example of onstat Status Line

Make a Final Backup

Use **ontape**, ON-Archive, or ON-Bar to make a level-0 backup and logical-log backup of your database server. Be sure to retain and properly label the tape volume that contains the backup. For more information about making backups, refer to the *Backup and Restore Guide for Informix Dynamic Server*.

Once you have made a level-0 backup, also perform a complete backup of the logical log, including the current logical-log file.

UNIX

If you use ON-Archive, execute the following command to make a full-system, level-0 backup:

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

♦

WIN NT

Do not use ON-Archive before you migrate to Windows NT. ♦

Bring the Source Database Server Off-Line

Execute the following command to take the source database server to off-line mode:

```
onmode -ky
```

Bring the source database server off-line to ensure that all common files are inactive. The source database server must be off-line because the target database server uses the same files. You cannot install the target database server if any of the files that it uses are active.

After you shut down the source database server, execute the following command to verify that it is off-line:

```
onstat -
```

Verify that you obtain the message `shared memory not initialized...` for off-line mode.

Important: *Make a final backup for each source database server instance that you plan to convert.*

UNIX

Reconfigure the UNIX Operating System

You might need to change some of the kernel parameters for your UNIX operating system before you install Universal Server or Dynamic Server with UD Option. To reconfigure the operating system, follow the directions in the machine-notes file included on your database server distribution media and the kernel-configuration instructions for your operating system. For information on the location of the machine-notes file, refer to "Documentation Notes, Release Notes, Machine Notes" on page 20 of the Introduction.

Install the Target Database Server

On UNIX platforms, you must be logged in as user **root** or on Windows NT platforms, you must be a member of the Informix-Admin group to install Universal Server or Dynamic Server with UD Option. Set the **INFORMIXDIR** environment variable to the directory where you plan to install Universal Server or Dynamic Server with UD Option.

Warning: *If you install the target database server in the same directory where the source database server resided, the installation script overwrites the older files. If you wish to preserve your source database server files, you must install the target database server in a different directory.*

Before you overwrite the source database server, you must take the following precautions:

- If you do not have the original media for the source database server, back up the **INFORMIXDIR** directory before you install Universal Server or Dynamic Server with UD Option.
- Copy the configuration files in the **etc** directory of **INFORMIXDIR** to another location on the file system.

Follow the directions in the *INFORMIX-Universal Server Installation Guide* to install Universal Server or Dynamic Server with UD Option. The installation script installs Universal Server or Dynamic Server with UD Option into the **INFORMIXDIR** directory specified for user **root** on UNIX or the Informix-Admin group on Windows NT to install Universal Server or Dynamic Server with UD Option. The installation script does not bring the target database server on-line.

WIN NT

For information on how to install Universal Server or Dynamic Server with UD Option on Windows NT, refer to the *INFORMIX-Universal Server Installation Guide* for Windows NT.♦

Verify That Environment Variables Are Set Correctly

After you install Universal Server or Dynamic Server with UD Option, ensure that the following environment variables, which are required, are set to the correct values:

- **INFORMIXSERVER**
- **ONCONFIG**
- **PATH**
- **INFORMIXSQLHOSTS** (if used)

Important: *The client application looks for the **sqlhosts** file or registry in the **etc** directory in the **INFORMIXDIR** directory. However, you can use the **INFORMIXSQLHOSTS** environment variable to change the location or name of the **sqlhosts** file.*

Update the ONCONFIG Configuration Files

You can customize your ONCONFIG configuration file and environment variables to take advantage of the new features introduced by Universal Server or Dynamic Server with UD Option. After you observe the performance of Universal Server or Dynamic Server with UD Option, you might want to make further adjustments.

For information on how to configure Universal Server or Dynamic Server with UD Option, refer to the *INFORMIX-Universal Server Administrator's Guide*. For information about environment variables, refer to the *Informix Guide to SQL: Reference*. For information about how to tune the configuration parameters, refer to the *INFORMIX-Universal Server Performance Guide*.

Important: *Use the same values for ROOTOFFSET, ROOTSIZE, and ROOTPATH that you used for OnLine Dynamic Server.*

UNIX

Add a Communications Support Module (Optional)

You can use either the default authentication policy or the Communications Support Module (CSM) with Universal Server or Dynamic Server with UD Option. After you install the CSM components, create entries in the **concsm.cfg** file and in the options field of the **sqlhosts** file to configure the CSM. For information on how to set up the CSM, refer to the *INFORMIX-Universal Server Administrator's Guide*.

Using Communications Support Modules with Client Applications

Existing client applications do not need to be recompiled or relinked if your Universal Server or Dynamic Server with UD Option does not use CSMs. If your Universal Server or Dynamic Server with UD Option uses a CSM, client applications must relink with new Informix libraries. The clients must also have a CSM installed and configured.

UNIX

Update the ON-Archive Configuration Files

If you use ON-Archive for your source database server backup and restore tool, and you will continue to use it with Universal Server or Dynamic Server with UD Option, you might need to update ON-Archive configuration parameters.

During the installation procedure for ON-Archive, the install script checks the **$INFORMIXDIR/etc** directory for files named **config.arc** and **oper_deflt.arc**. If the files do not exist, the install script provides them. If the files already exist, the install script does not overwrite the files. Instead, the install script provides additional files named **Config.arc** and **Oper_deflt.arc** (note the initial uppercase letters). Compare your current versions (**config.arc** and **oper_deflt.arc**) with the new versions and determine whether new or changed configuration parameters or qualifiers exist.

Bring the Target Database Server On-Line

Execute the following command to bring Universal Server or Dynamic Server with UD Option on-line for the first time:

```
oninit
```

As Universal Server or Dynamic Server with UD Option comes on-line for the first time, it modifies certain disk structures. This operation should extend the initialization process by only a minute or two. In the unlikely event that your disks cannot accommodate the growth in disk structures, you will find a message in the message-log file that instructs you to run **oncheck** on a table. The **oncheck** utility will tell you that you need to rebuild an index. You should rebuild the index as instructed.

Warning: *Universal Server or Dynamic Server with UD Option writes to the logical log with the transactions that result from creating the **sysmaster** database. If you run out of log space before the creation of the **sysmaster** database is complete, Universal Server* or *Dynamic Server with UD Option halts and indicates that you must back up the logical log. Once you have backed up the logical log, Universal Server or Dynamic Server with UD Option can finish building the **sysmaster** database.*

Check your Universal Server or Dynamic Server with UD Option message log for status messages that pertain to bringing Universal Server or Dynamic Server with UD Option on-line. For information about any messages in the message log, refer to the *INFORMIX-Universal Server Administrator's Guide*.

Important: *If the message file notes problems, solve the problems before you continue to the next step.*

Databases under Universal Server or Dynamic Server with UD Option contain almost twice as many system catalog tables than with OnLine Dynamic Server or Informix Dynamic Server. The combined size of the system catalog tables has grown about 2000 kilobytes per database. You may need to account for this growth if you have many databases and limited disk space.

Verify the Integrity of the Data

After Universal Server or Dynamic Server with UD Option finishes converting the system catalog tables, open each database with DB-Access and use **oncheck** to verify that no data was corrupted in the migration process. You can verify the integrity of the reserve pages, extents system catalog tables, data, and indexes, as shown in Figure 4-5.

Figure 4-5
Commands for Verifying the Data Integrity

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *database_name*
Check data	`oncheck -cD` *database_name*
Check indexes	`oncheck -cI` *database_name*
Check smart large objects	`oncheck -cs` *sbspase_name*
Check smart large object plus extents	`oncheck -cS` *sbspase_name*

Make an Initial Target Database Server Backup

Use your Universal Server or Dynamic Server with UD Option backup and restore tool (ON-Bar or **ontape**) to make a level-0 backup. Do not use ON-Archive. Do not overwrite the tapes you used earlier when you made your final backup of your source database server. If you use **ontape**, refer to the *INFORMIX-Universal Server Archive and Backup Guide*. If you use ON-Bar, refer to the *INFORMIX-Universal Server Backup and Restore Guide*.

Important: *Do not restore the backed up logical-log files from your source database server for your target database server.*

Migration Complete

When you finish the level-0 backup, the migration process is complete and users can use Universal Server or Dynamic Server with UD Option to access data safely.

Once you successfully migrate to Universal Server or Dynamic Server with UD Option, you might want to seek ways to obtain maximum performance. If you created sample queries for comparison, you can use them to characterize the performance differences between Informix Dynamic Server or OnLine Dynamic Server, Version 7.2, and Universal Server or Dynamic Server with UD Option. The results of these comparisons might suggest adjustments to configuration parameters, or to the layout of databases, tables, and chunks. For details on performance topics, refer to the *INFORMIX-Universal Server Performance Guide*.

Install and Configure Datablade Modules

After you have successfully migrated to Universal Server or Dynamic Server with UD Option, install and register any DataBlade modules, supplied by Informix or third-party vendors, that you want to add to Universal Server or Dynamic Server with UD Option. *Registration* is the process that makes the DataBlade module code available to use in a particular database. For more information on how to use DataBlade modules, refer to the *DataBlade Developers Kit User's Guide* and the *BladeManager User's Guide*.

Important: *Completely test Universal Server or Dynamic Server with UD Option with traditional relational data before you start to use DataBlade modules. After you successfully use DataBlade modules, you can begin to use extended data types, routines, and access methods.*

Reverting Universal Server or Dynamic Server with UD Option to OnLine Dynamic Server 7.2

Before you can revert to Version 7.2, remove all Universal Server or Dynamic Server with UD Option objects from the databases except those created by the boot scripts in the system catalog. Do not drop the objects that the **boot90.sql** and **boot901.sql** scripts created because the reversion utility uses them.

This section describes the steps for reverting from Universal Server or Dynamic Server with UD Option to OnLine Dynamic Server, Version 7.2. Complete the steps, described in the following sections:

1. Review the database schema to determine whether reversion is possible.
2. Save copies of the current configuration files.
3. Verify the integrity of the data.
4. Back up the Universal Server or Dynamic Server with UD Option database.
5. Remove Universal Server or Dynamic Server with UD Option features.
6. Run the reversion utility (**onmode -b**).
7. Modify configuration parameters.
8. Reset environment variables.
9. Remove CSM settings.
10. Start the desired version of your target database server.
11. Verify the integrity of the data.
12. Back up the target database server.
13. Return the target database server to on-line mode.

Determine Whether Reversion Is Possible

The easiest reversion scenario is a Universal Server or Dynamic Server with UD Option database that does not contain any new features. Run the reversion utility and modify the values of the configuration parameters.

Review the database schema to determine whether reversion to Informix Dynamic Server or OnLine Dynamic Server is possible. Does the schema file contain SQL statements that Informix Dynamic Server or OnLine Dynamic Server does not support? Does the database contain features that Informix Dynamic Server or OnLine Dynamic Server does not support, such as DataBlade modules, smart large objects, user-defined data types, and user-defined routines? Have any new SPL routines been created in Universal Server or Dynamic Server with UD Option?

To review the database schema, execute the **dbschema** utility. The **dbschema** syntax is described in Chapter 11, "The dbschema Utility." The following example displays complete information about the database, **db1**:

```
dbschema -d db1 -ss
```

Important: *You can revert from Universal Server or Dynamic Server with UD Option to Informix Dynamic Server or OnLine Dynamic Server only if you remove all Universal Server or Dynamic Server with UD Option features from the databases. If the databases contain Universal Server or Dynamic Server with UD Option features, reversion to Informix Dynamic Server or OnLine Dynamic Server will fail.*

UNIX

Save Copies of the Configuration Files

Save copies of the ONCONFIG and **concsm.cfg** files in case you decide to upgrade to Universal Server or Dynamic Server with UD Option again. Only Universal Server or Dynamic Server with UD Option uses the **concsm.cfg** file that is used to configure CSMs.

Verify the Integrity of the Data

Execute the following commands to check the integrity of the data:

```
oncheck -cI database_name
oncheck -cD database_name
oncheck -cr
oncheck -cc database_name
```

Back Up Your Source Database Server

Before you begin the reversion, make a complete backup. If you use **ontape** or ON-Archive, refer to the *INFORMIX-Universal Server Archive and Backup Guide*. If you use ON-Bar, refer to the *INFORMIX-Universal Server Backup and Restore Guide*.

Remove Universal Server or Dynamic Server with UD Option Features

Before you revert, you must remove Universal Server or Dynamic Server with UD Option features that OnLine Dynamic Server or Informix Dynamic Server do not support:

- DataBlade modules
- User-defined routines and user-defined functions
- Indexes over 255 bytes
- Extended data types
- Smart large objects
- Features specific to Universal Server or Dynamic Server with UD Option
- Stored procedures created under Universal Server or Dynamic Server with UD Option
- Secondary access methods
- Virtual tables in extspaces
- Sbspaces and extspaces

Uninstall Detachable Modules

Use BladeManager to unregister all DataBlade modules. When you unregister DataBlade modules, you remove all data types and routines that the DataBlade modules define.

UNIX

BladeManager is a command-line utility (**blademgr**) stored in the **$INFORMIXDIR/bin** directory. ♦

WIN NT

BladeManager is a command-line utility (**blademgr**) stored in the **%INFORMIXDIR%\bin** directory. ♦

For more information, refer to the *BladeManager User's Guide*.

However, a few hidden tables and error messages remain in each database that BladeManager connects to. These tables contain the list of DataBlade modules and DataBlade module interfaces installed in the database. The hidden table names are **sysbldregistered**, **sysbldirequired**, **sysbldiprovided**, **sysbldobjects**, and **sysbldobjdepends**. The error messages are in the **syserrors** system catalog table and have the **sqlstate** field beginning with **UGENx**.

After you unregister the DataBlade modules, execute the following SQL script to delete the hidden tables and error messages from each database:

```
DROP TABLE sysbldregistered; --All DataBlades registered
DROP TABLE sysbldirequired;  --Inter-DataBlade dependencies
DROP TABLE sysbldiprovided;  --DataBlade interfaces
DROP TABLE sysbldobjects;   --All objects created by DataBlade
DROP TABLE sysbldobjdepends; --Dependencies between objects
DELETE FROM syserrors WHERE sqlstate LIKE ‘UGEN_’;
```

Remove User-Defined Routines and User-Defined Functions

Remove all user-defined routines and functions created in Universal Server or Dynamic Server with UD Option because Informix Dynamic Server or OnLine Dynamic Server do not support them. For information on user-defined routines, refer to *Extending INFORMIX-Universal Server: User-Defined Routines*.

Drop Indexes over 255 Bytes

In Version 7.2 of the database server, the maximum key length of an index is 255 bytes. For Universal Server or Dynamic Server with UD Option, the maximum key length of an index is 390 bytes. If your indexes have keys longer than 255 bytes, you must drop them before you revert the database.

Remove Extended Data Types and Smart Large Objects

Universal Server or Dynamic Server with UD Option provides many data types that Informix Dynamic Server or OnLine Dynamic Server do not support. Before you revert, drop tables, columns, views, and indexes that contain the following data types:

- Built-in data types not available in Informix Dynamic Server or OnLine Dynamic Server:
  - BOOLEAN
  - INT8
  - LVARCHAR
  - SERIAL8
- Smart large objects (CLOB and BLOB)
- User-defined data types (OPAQUE and DISTINCT)
- Collection data types (SET, MULTISET, LIST)
- Row data types (ROW)
- Data types provided by DataBlade modules

If you do not want to drop the tables, columns, or views, you can change the data to a type that Informix Dynamic Server or OnLine Dynamic Server supports. For example, if the database contains one new table with user-defined data types, either delete that table, or change the user-defined data types to legacy data types and drop the user-defined data types from the system catalog before you revert.

For information on Universal Server or Dynamic Server with UD Option data types, refer to the *Informix Guide to SQL: Reference* and *Extending INFORMIX-Universal Server: Data Types*.

Remove Features Specific to Universal Server or Dynamic Server with UD Option

Before you revert, you must remove features that are specific to Universal Server or Dynamic Server with UD Option from client applications and the databases. For example, Informix Dynamic Server or OnLine Dynamic Server does not support operators and casts to any data type. For information, refer to the *Informix Guide to SQL: Syntax*, *Extending INFORMIX-Universal Server: Data Types*, and the *INFORMIX-ESQL/C Programmer's Manual*.

Remove Stored Procedures Created in Universal Server or Dynamic Server with UD Option

Before you revert, drop all stored procedures that are created in Universal Server or Dynamic Server with UD Option. Informix Dynamic Server, OnLine Dynamic Server, Universal Server, and Dynamic Server with UD Option support stored procedures. However, the internal structure of a stored procedure created in Universal Server or Dynamic Server with UD Option is not backward-compatible and does not run under Informix Dynamic Server or OnLine Dynamic Server. For information on stored procedures, refer to the *Informix Guide to SQL: Syntax*.

Drop Secondary Access Methods (Indexes)

Drop the following secondary access methods (indexes) that Informix Dynamic Server or OnLine Dynamic Server does not support:

- Generic B-tree indexes on user-defined and built-in data types
- R-tree indexes on spatial data such as maps and diagrams
- Functional indexes on values returned from user-defined functions
- User-defined indexes that a DataBlade module (such as Excalibur Text DataBlade) provides

For more information on indexes, refer to the *INFORMIX-Universal Server Performance Guide*.

Remove Virtual Tables in External Spaces

Drop all virtual tables stored in external spaces and remove the access methods used to access the external data. For more information on primary access methods, refer to the *Virtual-Table Interface Programmer's Manual*.

Remove Sbspaces and External Spaces

You should already have deleted columns that contain smart large objects (BLOB and CLOB) and all virtual tables. Now delete all sbspaces and extspaces. To delete these spaces, execute the following command where *spacename* is the name of the sbspace or extspace:

```
onspaces -d spacename
```

For more information on sbspaces and external spaces, refer to the *INFORMIX-Universal Server Administrator's Guide*.

Run the Reversion Utility

Universal Server or Dynamic Server with UD Option must be running when you execute the reversion utility. The reversion utility detects and lists remaining features that are specific to Universal Server or Dynamic Server with UD Option that you should remove before reversion can complete.

Execute the reversion utility to revert a Universal Server or Dynamic Server with UD Option database to OnLine Dynamic Server, Version 7.2:

```
onmode -b 7.2
```

After the reversion is complete, Universal Server or Dynamic Server with UD Option is off-line. The reversion utility drops the Universal Server or Dynamic Server with UD Option system catalog tables and restores compatibility so that users can access the data with OnLine Dynamic Server. The reversion utility does not revert changes made to the layout of the data that do not affect compatibility.

You may also revert to another version of the database server, such as Version 7.22, Version 7.24 or Version 7.3, Version 7.3. For more information about the **onmode -b** command, refer to "The onmode Utility" on page 11-69.

Modify Configuration Parameters

Remove the following configuration parameters that Informix Dynamic Server or OnLine Dynamic Server does not support:

- SBSPACENAME
- VPCLASS

You might also need to adjust the values of existing configuration parameters. Alternatively, you can replace the Universal Server or Dynamic Server with UD Option ONCONFIG file with the Informix Dynamic Server or OnLine Dynamic Server ONCONFIG file that you used before you upgraded.

Reset Environment Variables

Reset the environment variables to values that are appropriate for Informix Dynamic Server or OnLine Dynamic Server. Also remove the following environment variables that Informix Dynamic Server or OnLine Dynamic Server does not support:

- INFORMIXCONCSMCFG
- INFORMIXKEYTAB

UNIX

Remove Communications Support Module Settings

If your Universal Server or Dynamic Server with UD Option instance uses CSMs, remove the **csm** option settings from the **sqlhosts** entries for the database server. Otherwise the older database server will return an invalid **sqlhosts** options error. Also delete the **concsm.cfg** file.

Install and Start the Target Database Server

Install and configure the target database server according to the instructions in your *Administrator's Guide*.

Execute the **oninit -s** command to bring your target database server to quiescent mode.

Verify the Integrity of the Data

Before you allow users to access the databases, check the integrity of the data. Follow the steps described in "Verify the Integrity of the Data" on page 4-13.

Back Up the Target Database Server

After you complete the reversion, make a level-0 backup. Use either ON-Archive or the **ontape** utility to make the backup. For information about ON-Archive and **ontape**, refer to your *Backup and Restore Guide*.

Important: *Do not overwrite the tapes that you used to back up your source database server.*

Return the Target Database Server to On-Line Mode

To bring Informix Dynamic Server or OnLine Dynamic Server on-line, execute the **onmode -m** command. The reversion is now complete, and users can access the converted data.

Upgrading Universal Server 9.10 to Dynamic Server with UD Option 9.13

You can upgrade to a later version of Universal Server (such as Version 9.12 to Universal Server or Dynamic Server with UD Option, Version 9.13) automatically when you first access your database. No specific action is required.

After you upgrade, you can use **oncheck** to verify the integrity of the reserve pages, extents, system catalog tables, data, indexes, and sbspaces.

Figure 4-6
Commands for Verifying the Data Integrity

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *database_name*
Check data	`oncheck -cD` *database_name*
Check indexes	`oncheck -cI` *database_name*
Check smart large objects	`oncheck -cs` *sbspase_name*
Check smart large object plus extents	`oncheck -cS` *sbspase_name*

Rebuilding R-Tree Indexes

The disk organization of R-tree indexes changed in Universal Server, Version 9.11. If you are upgrading from Universal Server, Version 9.10, to Version 9.11 or higher, you must drop and re-create your R-tree indexes. Dropping and recreating R-tree indexes gives you larger indexes with faster searching capabilities.

Tables Containing Collections

If a table column has a collection data type, you must unload the collection data before you migrate with INFORMIX-Universal Server or Informix Dynamic Server with Universal Data Option and then reload the data after the migration.

Chapter

5

Migrating the Database Server 6.0 or Later to 7.3

This chapter describes the procedures to migrate between Informix Dynamic Server, Version 7.3, and versions 6.0 and later.

You can use the same procedure to upgrade to Version 7.2, 7.22, 7.23, or 7.24, if you note the version-specific features and any necessary additional steps. When you upgrade database server versions, you might want to upgrade database applications as well to take advantage of new features. The release-notes files that accompany the software describe new features.

This chapter covers the following topics:

- Changes introduced by versions 6.0 to 7.3 of the database server
- Preparing to migrate
- Upgrading your database server to Informix Dynamic Server, Version 7.3
- Installing Informix Enterprise Command Center
- Reverting your database server to an earlier version

You can upgrade from or revert to the following versions:

WIN NT

- OnLine Dynamic Server, Version 7.12 ♦
- OnLine Dynamic Server, Version 6.0, Version 7.10UCx, Version 7.10UD1 (UNIX only) through Version 7.14, and Version 7.2 through Version 7.24.

Tip: *For information on how to upgrade from pre-Version 6.0 versions of your database server, refer to Chapter 6, "Migrating Pre-6.0 to Informix Dynamic Server 7.3." For information on how to move the database server data between UNIX and Windows NT environments, refer to Chapter 9, "Moving Between Database Servers."*

Changes Introduced in Database Server Version 6.0 to Version 7.3

This section describes changes to environment variables, configuration parameters, and new database server functionality that affects migration that was introduced by different Informix database server versions. The changes are described in descending order, with the changes to the database server, Version 7.3, discussed first. For each version, the environment variables and configuration parameters are discussed and new features or changes to existing features are listed for that version.

Environment Variables Introduced in Version 7.3

Informix Dynamic Server, Version 7.3, introduces several new environment variables and maintains several otherwise obsolete environment variables for backward compatibility. These new environment variables are described in the *Informix Guide to SQL: Reference*.

- INFORMIXKEYTAB
- IFX_DIRECTIVES

Informix Dynamic Server, Version 7.3, includes the Informix Storage Manager (ISM), a backup and recovery media management utility, with new environment variables. For more information, refer to *Informix Storage Manager Administrator's Guide*.

Configuration Parameters Introduced in Version 7.3

Informix Dynamic Server, Version 7.3, introduces several new configuration parameters. The CDR configuration parameters are described in the *Guide to Informix Enterprise Replication*. The ISM variables are described in the *Informix Storage Manager Administrator's Guide*. The optimizer configuration parameters are described in your *Performance Guide*. The other parameters are described in your *Administrator's Guide*.

UNIX

- CDR_DSLOCKWAIT
- CDR_EVALTHREADS

- CDR_LOGBUFFERS
- CDR_QUEUEMEM
- DIRECTIVES (Optimizer directives)
- ISM_DATA_POOL
- ISM_LOG_POOL
- OPT_GOAL
- RESTARTABLE_RESTORE (Defaults to Off)
- SYSALARMPROGRAM
- TBLSPACE_STATS ♦

WIN NT

- LOG_BACKUP_MODE
- OPTICAL_LIB_PATH (For new optical functionality)
- STAGEBLOB (New for Windows NT) ♦

Features Introduced in Version 7.3

Informix Dynamic Server, Version 7.3 introduces several features that affect migration. These features include new built-in functions to the database server, changes to backup and restore mechanisms, and multiple residency options on Windows NT. For Informix Dynamic Server, Version 7.3, the Informix Enterprise Command Center interface is available on Windows NT and UNIX.

New Functions

The following new built-in functions are available in Informix Dynamic Server, Version 7.3, to facilitate migration to Informix database servers:

- DECODE
- NVL
- TO_DATE
- TO_CHAR
- LPAD
- RPAD
- SUBSTR
- SUBSTRING
- REPLACE

The names of these functions are reserved words in the database server. For more information on these functions, refer to the *Informix Guide to SQL: Syntax*.

Migrating ontape and onarchive to ON-Bar

Informix Dynamic Server, Version 7.3 includes **ontape**, ON-Archive, and ON-Bar for backup and recovery. If you migrate from previous versions of the database server, you may want to upgrade from **ontape** or **onarchive** to ON-Bar. In Version 7.3, ON-Bar uses a new Informix Storage Manager (ISM) to read or write data from tape or disk. The ISM is administered through the ISM Administrator, which you can start from Informix Enterprise Command Center.

Important: *To restore data, always use the same backup and restore method you used to create your backup.*

For more information on ON-Bar, refer to the *INFORMIX-OnLine Dynamic Server Backup and Restore Guide*. For more information on ISM, refer to the *Informix Storage Manager Administrator's Guide*.

WIN NT

Multiple Residency and High Availability

As of Informix Dynamic Server, Version 7.3, *multiple residency* is available on Windows NT. Multiple residency allows multiple database servers and their associated shared memory and disk structures to coexist on a single computer. Each instance of the database server has its own **INFORMIXSERVER**, **INFORMIXDIR**, and **ONCONFIG** values.

In Informix Dynamic Server, Version 7.3 on Windows NT, multiple residency enables the database server to be installed as a cluster-aware application, which in turn can facilitate high availability. A cluster-aware application is an application that is registered with the resource manager and provides dynamically linked libraries (resource DLLs).

Tip: *Prior database server versions cannot co-exist with Informix Dynamic Server, Version 7.3, since multiple residency is only available as of Version 7.3. If a prior version is present, you must uninstall and re-install the new version to migrate on Windows NT. The installation process uses configuration files from the previous version. For more information on how to install Version 7.3 on Windows NT, see your "Installation Guide."*

What Is High Availability?

High availability provides redundant components in a cluster of two or more nodes at an operating-system level. Whenever a failure occurs on one node (one Windows NT computer), the cluster manager restarts the failed application on the surviving node. The cluster manager and resource manager detect failures and communicate to other subsystems and applications.

Tip: *When you upgrade database server versions, you might need to reinstall backups of the target database server, to ensure that an entire cluster is upgraded.*

For more information on multiple residency and high availability, refer to your *Administrator's Guide*.

Environment Variable Changes in Version 7.2x

OnLine Dynamic Server, Version 7.2x, introduces several new environment variables and maintains several otherwise obsolete environment variables for backward compatibility.

Environment Variables Introduced in Version 7.2x

Figure 5-1 shows new environment variables for Version 7.2x. Review the descriptions of these environment variables to determine whether you need to set them. The **Reference** column in Figure 5-1 indicates the manuals that provide information about these environment variables. Figure 5-1 uses the following abbreviations for Informix manuals:

- HPL: *Guide to the High-Performance Loader*. (The High-Performance Loader (HPL) is available on Windows NT as of Version 7.3.)
- GLS: *Informix Guide to GLS Functionality*
- ESQL/C: *INFORMIX-ESQL/C Programmer's Manual*
- REF: *Informix Guide to SQL: Reference*

Figure 5-1
Environment Variables Introduced in Version 7.2x

Environment Variable	Variable Affects	Reference
CC8BITLEVEL	ESQL/C only	GLS
CLIENT_LOCALE	Client applications only	GLS
DBCENTURY	SQL APIs only	REF, GLS
DBONPLOAD	High-Performance Loader only	REF, HPL
DB_LOCALE	Database locale	GLS
ESQLMF	ESQL/C compilation	GLS
GLS8BITFSYS	8-bit clean	GLS
GL_DATE	Date format	GLS
GL_DATETIME	Time format	GLS
IFX_AUTOFREE	ESQL/C compilation	ESQL/C
NODEFDAC	Default privileges	REF
ONPLOAD	High-Performance Loader	REF, HPL
OPTOFC	ESQL/C compilation	ESQL/C
PLCONFIG	High-Performance Loader	REF, HPL
SERVER_LOCALE	Database server locale	GLS
THREADLIB	ESQL/C only	REF, ESQL/C

Tip: *The IFX_AUTOFREE environment variable introduced in Version 7.22 is discontinued in future releases. If you want to take advantage of this feature, use SET AUTOFREE syntax in your application.*

NLS

Environment Variables Maintained for Backward Compatibility

OnLine Dynamic Server, Version 7.2x, supports the environment variables in the following list for backward compatibility with earlier Informix products. If you do not have databases and applications from pre-7.2 versions, you would not use these environment variables. These environment variables are described in Version 7.1 of the *Informix Guide to SQL: Reference*. The interaction of these environment variables with variables that control GLS is described in the *Informix Guide to GLS Functionality*. The following environment variables maintain backward compatibility:

- **COLLCSHAR**
- **DBAPICODE**
- **DBNLS**
- **LANG**
- **LC_COLLATE**
- **LC_CTYPE**
- **LC_MONETARY**
- **LC_NUMERIC**
- **LC_TIME**

Configuration Parameter Changes in Version 7.2x

OnLine Dynamic Server, Version 7.2x, includes new configuration parameters that might affect your installation. You might also need to adjust the values of existing parameters. These configuration parameters are described in your *Administrator's Guide*.

Configuration Parameters Introduced in Version 7.22

Enterprise Replication (and Workgroup Replication for OnLine Workgroup Server, Version 7.2x, and OnLine Dynamic Server, Version 7.2x, on Windows NT only) use the following new configuration parameters:

- CDR_LOGBUFFERS
- CDR_EVALTHREADS
- CDR_DSLOCKWAIT
- CDR_QUEUEMEM
- CDR_NIFUSEHELP
- CDR_NIFMEMS
- CDR_NIFQUEUES

For more information on these configuration parameters, see the *Guide to Informix Enterprise Replication*.

Configuration Parameter Changes in Version 7.22

The default value of ALARMPROGRAM has changed.

Configuration Parameters Introduced in Version 7.2

Version 7.2 and later uses the following new configuration parameters:

- BUFFERS (new definition)
- HETERO_COMMIT
- MAX_PDQPRIORITY (new definition)

Configuration Parameters Dropped in Version 7.2

Version 7.2x does not use the following configuration parameters:

- BUFFSIZE
- PDQPRIORITY (the default value is always zero)

If you do not set PDQPRIORITY through the environment variable or SQL statement, PDQ is turned off for queries (PDQPRIORITY = 0). For information on where to set environment variables, refer to the *Informix Guide to SQL: Reference*.

In-Place ALTER TABLE Introduced in Version 7.24 and Later

OnLine Dynamic Server, Version 7.24, introduced an In-Place option to the ALTER TABLE statements that was not possible in previous versions. With the In-Place modifier, you can change a table without creating a duplicate copy of the table. For reversion procedures to pre-Version 7.24, see "Modify In-Place ALTER TABLE (Version 7.24 and Later)" on page 5-37. For more information on In-Place ALTER TABLE statements, see your *Performance Guide*.

Enhancements to sqlhosts File or Registry in Version 7.23 and Later

INFORMIX-OnLine Dynamic Server, Version 7.23, introduces formatting changes and new syntax options to the **sqlhosts** file or registry. The fifth field, the **options** field, was modified to accommodate new syntax options.

The following list is a review of the **sqlhosts** or registry fields:

```
FIELD 1        FIELD 2  FIELD 3   FIELD 4       FIELD 5
dbservername   nettype  hostname  servicename   options
```

The **options** field can contain columns separated by a comma or white space that represents the end of the column. Client and database server applications check each column to determine whether the option is supported in the database server release.

In earlier versions, the **options** field could contain only a single character. In Version 7.23 and later, longer names are supported. Earlier versions, such as Version 7.1, cannot process the longer syntax.

Tip: *If you maintain more than one version of the database server on UNIX, use separate* ***sqlhosts*** *file entries for each version. Alternatively, you can use separate entries with an alias to the appropriate database server. You cannot maintain two versions of the database server on Windows NT.*

Figure 5-2 lists the OnLine Dynamic Server, Version 7.23 and later, **sqlhosts** file or registry components.

Figure 5-2
sqlhosts File or Registry Fields

dbservername	nettype	hostname	servicename	options
`dbservername.server1`	`olsoctcp`	`clipper`	`14000`	`k=1,r=1 b=2000`

In the sample **sqlhosts** file, the **options** field contains three options contained in three columns:

Column	Option
Column 1	`k=1`
Column 2	`r=1`
Column 3	`b=2000`

Important: *Informix recommends that you use field 5,* ***options****, in Version 7.23 and later, for the following options only:* ***b****,* ***k****,* ***r****,* ***s****. If you do not want any of these options but do want other options, use* `k=1` *in column 5, which is the default. Place other options in subsequent columns.*

For more information on the components of the **sqlhosts** file or registry, or on how to define two **sqlhosts** files or registries, see your *Administrator's Guide*.

Global Language Support Changes in Version 7.2x

NLS

OnLine Dynamic Server, Version 6.0, introduced Native Language Support (NLS). NLS supports single-byte locales, but not multibyte locales. ♦

WIN NT

OnLine Dynamic Server for Windows NT, Version 7.12, supports NLS. ♦

GLS

Informix Version 7.2x products use Global Language Support (GLS). GLS lets the database sever, Version 7.2x and later, handle different languages, cultural conventions, and code sets for Asian, European, Latin American, and Middle Eastern countries. Chapter 10, "Changing Locales," discusses the migration implications of GLS. The *Informix Guide to GLS Functionality* provides a full description of GLS. ♦

UNIX

Environment Variable Changes in Version 7.10.UD1

OnLine Dynamic Server, Version 7.10.UD1, introduced the following new environment variables:

- **INFORMIXOPCACHE**
- **INFORMIXSQLHOSTS**
- **NODEFDAC**
- **OPTCOMPIND** (new definition)
- **PSORT_NPROCS** (new definition)

Configuration Parameter Changes in Version 7.10.UD1

OnLine Dynamic Server, Version 7.10.UD1, introduced several new configuration parameters and dropped several others.

Configuration Parameters Introduced in Version 7.10.UD1

OnLine Dynamic Server, Version 7.10.UD1, introduced the following configuration parameters:

- LBUPRESERVE
- ONDBSPACEDOWN
- OPCACHEMAX
- OPTCOMPIND (new default value)

Configuration Parameters Dropped in Version 7.10.UD1

OnLine Dynamic Server, Version 7.10.UD1, dropped the following configuration parameters. OnLine Dynamic Server allocates resources dynamically for the structures that these parameters controlled in previous releases:

- CHUNKS
- DBSPACES
- USERTHREADS
- TBLSPACES
- TRANSACTIONS

You might need to reset the value of the LOCKS configuration parameter because it previously depended on the value of TRANSACTIONS.

Configuration Parameters Moved in Version 7.10.UD1

OnLine Dynamic Server, Version 7.10.UD1, moved the following configuration parameters from the ONCONFIG file.

- ADTPATH
- ADTSIZE
- ADTERR
- ADTMODE

UNIX

In the UNIX environment, these configuration parameters are in the audit configuration file (**$INFORMIXDIR/aaodir/adtcfg.std**) ♦

WIN NT

In the Windows NT environment, these configuration parameters are in the audit configuration file (**%INFORMIXDIR%\aaodir\adtcfg**). ♦

Changes to Blobspace Requirements in Version 7.10.UD1

Versions of OnLine Dynamic Server before 7.10.UD1 marked a partition blob page as full if the page was more than one-third full. Version 7.10.UD1 uses a threshold of one-half of the page size.

In cases where partition blobs have a random size, both schemes use about the same amount of disk space. However, in certain situations the required disk space changes. If you have many partition blobs that are just larger than one-third of a page but less than one-half of a page, the new scheme reduces your space requirements by a factor of two. On the other hand, if you have partition blobs that are just less than one-third of a page and others that are just less than two-thirds of a page, you might see an increase in disk requirements of about 33 percent.

Environment Variable Changes in Version 7.1

OnLine Dynamic Server, Version 7.1, introduced the following environment variables:

- **DELIMIDENT**
- **FET_BUF_SIZE**

Configuration Parameter Changes in Version 7.1

OnLine Dynamic Server, Version 7.1, introduced the following configuration parameters:

- ALARMPROGRAM
- DATASKIP
- DS_MAX_QUERIES
- DS_MAX_SCANS
- DS_TOTAL_MEMORY
- MAX_PDQPRIORITY
- OPTCOMPIND

In addition, the PDQPRIORITY configuration parameter was introduced in Version 7.1, but dropped in Version 7.2x.

In Version 7.1, the default values for the configuration parameters LTXHWM and LTXEHWM changed from 80 and 90 to 50 and 60, respectively. OnLine Dynamic Server initialization provides a warning if your ONCONFIG file contains values for these parameters greater than 50 and 60.

UNIX

ON-Archive Changes in Version 7.1

The names of the ON-Archive error message and help files changed between Version 6.0 and Version 7.1. If you use a **config.arc** file from Version 6.0, you must change the filenames in **$INFORMIXDIR/etc/config.arc**. If you use the default **Config.arc** file (note the initial uppercase letter) that is installed with Version 7.1, you do not need to make any changes. For more information, refer to the *INFORMIX-OnLine Dynamic Server Archive and Backup Guide*.

Environment Variable Changes in Version 6.0

Version 6.0 of OnLine Dynamic Server introduced name changes for environment variables and utilities. Environment variable names that began with **TB** in earlier versions begin with **ON** in OnLine Dynamic Server 6.0 or later. For instance, the **TBCONFIG** environment variable was replaced by the **ONCONFIG** environment variable.

Utility names that began with **tb** in earlier versions begin with **on** in OnLine Dynamic Server 6.0 and later. For instance, **tbcheck** was replaced by **oncheck**. For a complete list of utilities, refer to your *Administrator's Guide*.

If you have not already changed the names of these environment variables and utilities in your shell scripts and in the login files of your users, make the changes now.

Preparing to Migrate Between Versions

When you migrate from one version of the database server to another, Informix suggests that you follow these guidelines:

- Review the release notes for the version of your database server for information about new features, installation, and fixes to problems. Modify applications as needed.

  The release notes are in one of the following directories:

UNIX

- **$INFORMIXDIR/release/en_us/0333**. ♦

WIN NT

- **%INFORMIXDIR%\release\en_us\0333 (**as of Version 7.2)

  Release Notes appear in the Informix folder. To display this folder, choose **Start→Programs→Informix** from the Task Bar. ♦

- Retain both versions of the Informix product software on disk, if possible (if you have enough disk resources). On Windows NT, you can retain multiple versions of the Informix product as of Version 7.3 of the database server.
- Check the documentation notes for information about features not covered in the manuals.
- Retain the installation media from both versions of the Informix product software.
- Make a level-0 backup of the database server before and after migration.

Before you upgrade, migrate, or revert your database server, complete the following steps, which are described in the next sections:

1. Install the latest maintenance release for the current version.
2. Check available space and system requirements.
3. Save copies of the current configuration files.
4. Shutdown your database sever.s.
5. Verify the integrity of the data.
6. Back up your database server files.

Important: *Repeat steps 3 through 6 for each instance of your database server that you are migrating.*

Install the Latest Maintenance Release for the Current Version

Informix recommends that you install the latest maintenance release for your current database server version before you migrate to a new version, especially if you use Version 5.x or 6.x. In this scenario, you plan to migrate from INFORMIX-OnLine 5.03 to Informix Dynamic Server, Version 7.3, on UNIX.

First, install the latest maintenance release for INFORMIX-OnLine, then migrate to Informix Dynamic Server,Version 7.3. Many minor changes to the 5.x versions are also in the 7.x versions.

For additional information, refer to the installation guide for your database server and the chapters on installation and configuration in your *Administrator's Guide*.

Check Available Space and System Requirements

The database server requires 1100 free pages of logical-log space (around 2000 kilobytes) to build the **sysmaster** database. The Informix Dynamic Server, Version 7.3, media is more than twice as large as previous versions because of extra GLS files.

Before you install your database server and the administration tool, Informix Enterprise Command Center, verify that your system meets the minimum space and hardware requirements. Informix Dynamic Server, Version 7.3, requires 15 to 20 percent more space than the previous version because it includes administration tools, such as Informix Enterprise Command Center and Relational Object Manager.

WIN NT

The database server runs on Windows NT 4.0 with Service Pack 3 on an NTFS drive. The administration tools run on Windows NT 3.51, Windows NT 4.0, and Windows 95 on either a FAT or NTFS drive.

For information on the system requirements, refer to the **read_ods.txt** file in Answers OnLine, version 1.7 or earlier.

Save Copies of the Current Configuration Files

Save copies of the configuration files for each instance of your database server. Keep the copies available to use later. Save the configuration files that Figure 5-3 lists, if they exist.

Figure 5-3
ODS or IDS Configuration Files

UNIX	Windows NT
$INFORMIXDIR/etc/$ONCONFIG	**%INFORMIXDIR%\etc\ONCONFIG**
$INFORMIXDIR/etc/onconfig	**%INFORMIXDIR%\etc\onconfig**
$INFORMIXDIR/etc/onconfig.std	**%INFORMIXDIR%\etc\onconfig.std**
$INFORMIXDIR/aaodir/adtcfg *	**%INFORMIXDIR%\aaodir\adtcfg.***
$INFORMIXDIR/dbssodir/adtmasks *	**%INFORMIXDIR%\dbssodir\adtmasks.***
$INFORMIXDIR/etc/sqlhosts	
$INFORMIXDIR/etc/termcap	
$INFORMIXSQLHOSTS	

UNIX

If you use ON-Archive to back up and restore your database server and the logical logs, you must also copy and save the configuration files in the following list:

- **$INFORMIXDIR/etc/$ARC_CONFIG**
- **$INFORMIXDIR/etc/config.arc**
- **$INFORMIXDIR/etc/oper_deflt.arc**
- **$INFORMIXDIR/etc/tctermcap** ♦

If you use ON-Bar to back up and restore your database server and the logical logs, you must also copy and save the configuration files in **$INFORMIXDIR/etc/ixbar.***.

Shut Down the Database Server

Communicate to client users how long you expect the database server to be off-line for the migration. Terminate all database server processes, and place your database server in quiescent mode (also called *administration mode*).

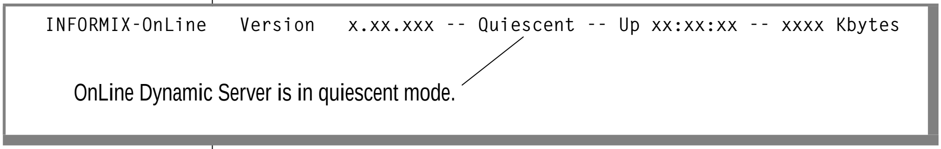
To shut down the database server gracefully

1. Warn all users that you plan to shut down the database server and wait for them to exit.
2. Become user **informix** on UNIX platforms, or on Windows NT, you must be a member of the **Informix-Admin** group.
3. Execute the following command to take the database sever to quiescent mode.

   ```
   onmode -sy
   ```

4. Wait until your database server is in quiescent mode.

   To verify the mode of your database server, execute the **onstat -** command. The first line of the **onstat** output contains the status of your database server. Figure 5-4 shows that the database server is in quiescent mode.



```
INFORMIX-OnLine   Version   x.xx.xxx -- Quiescent -- Up xx:xx:xx -- xxxx Kbytes
```

OnLine Dynamic Server is in quiescent mode.

Figure 5-4
Example of onstat Status Line

5. Execute the following command to force a new logical log:

   ```
   onmode -l
   ```

6. Execute the following command to force a checkpoint:

   ```
   onmode-c
   ```

7. Execute the following command to shut down the database server:

   ```
   onmode -yuk
   ```

Tip: *Monitor your log activity to verify that all commands were executed properly and to check for inconsistencies prior to migration.*

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of data before you make a level-0 (complete) backup. If you find any problems with the data, fix them before you make the backup. You can verify the integrity of the reserve pages, extents, system catalog tables, data, and indexes. First, you need to obtain a list of the databases on your database server.

Figure 5-5 lists the **oncheck** commands that verify data integrity. Informix recommends that at a minimum you run the **-cr** and **-ce** options.

Figure 5-5
Commands for Verifying the Data Integrity

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *database_name*
Check data	`oncheck -cD` *database_name*
Check indexes	`oncheck -cI` *database_name*

For information on **oncheck**, refer to your *Administrator's Guide*.

Back Up Your Database Server Files

Use your preferred backup method (**ontape**, ON-Archive, or ON-Bar) to make a complete (level-0) backup of each database server that you plan to migrate.

WIN NT

The Windows NT environment does not support ON-Archive. ♦

For more information about how to use **ontape**, ON-Archive, or ON-Bar to back up your database server, refer to your *Backup and Restore Guide*.

Warning: *Backups that you make under older versions of your database server are not compatible with the newer version. Do not try to restore these backups to the newer version.*

Preparation Complete

Now you are ready to upgrade, migrate, or revert your database server. For information on how to upgrade, see "Changing Database Server Definitions" on page 5-32. For information on how to revert, see "Reverting to OnLine Dynamic Server" on page 5-33.

Upgrading to Informix Dynamic Server, Version 7.3

This section describes how to upgrade OnLine Dynamic Server, Version 6.0 or later, to Informix Dynamic Server, Version 7.3. You should have completed the preparatory steps, described in "Preparing to Migrate Between Versions" on page 5-18.

When you upgrade, you can install and test Informix Dynamic Server, Version 7.3, with the same database server name, configuration files, environment variables, and **sqlhosts** information or registry information that you used for the earlier version. After you install Informix Dynamic Server, Version 7.3, and verify that it works, you might want to modify the configuration files and environment variables to take advantage of new features.

When you upgrade your database server, complete the following steps, which are described in more detail in the sections that follow:

1. Save an output file of SQL statements for access paths
2. Bring the source database server off-line.

UNIX

3. Reconfigure the operating system, if necessary. ♦
4. Install the target version of your database server.
5. Install the INFORMIX-Enterprise Command Center (Version 7.3 only).
6. Customize the database server environment.
7. Update the ONCONFIG configuration file.

UNIX

8. Update the ON-Archive configuration files. ♦
9. Configure the database server for Enterprise Replication, if desired.

WIN NT

10. Install and configure **onsnmp**. ♦
11. Bring the target database server on-line.

12. Run UPDATE STATISTICS.
13. Verify the integrity of the data.
14. Check space requirements for BYTE and TEXT data.
15. Make an initial backup of your database server.
16. Verify the access path of your SQL statements.

Save an Output File of SQL Statements for Access Paths

Save a file of output from any SET EXPLAIN statements, so that later you can verify that access paths of your SQL statements do not change when you migrate to your target database server. SET EXPLAIN writes the access path that the optimizer chooses for each query to the SET EXPLAIN output file. The optimizer chooses the fastest path of execution for table joins.

WIN NT

For Windows NT, the SET EXPLAIN output file name is **%INFORMIXDIR%\sqlexpln\*username*.out**. ♦

Bring the Source Database Server Off-Line

Shut down your database server to ensure that all common files are inactive.

The database server must be off-line because the older and the newer versions share common files. You cannot install the database server if any of the common files are active.

WIN NT

The installation program automatically shuts down the old database server and starts the new database server. ♦

UNIX

Reconfigure the Operating System

You might need to change some of the kernel parameters for your UNIX operating system before you install Informix Dynamic Server, Version 7.3. To reconfigure the operating system, follow the directions in the machine-notes file included on your Informix Dynamic Server, Version 7.3, distribution media and the kernel-configuration instructions for your operating system.

Install the Target Database Server

On UNIX platforms, you must be logged in as user **root,** or on Windows NT platforms, you must be a member of the **Administration** group to install your database server. Set the **INFORMIXDIR** environment variable to the directory where you plan to install your database server.

UNIX

The installation script installs your database server into the **INFORMIXDIR** directory specified for user **root.** The installation script does not bring the database server on-line. ♦

WIN NT

The setup program installs and brings up the database server on Windows NT. Follow the directions in your *Installation Guide* to install your database server. ♦

Warning: *If you install your database server in the same directory where the earlier version of database server resides, the newer version overwrites the older files. If you wish to preserve the files for the earlier version, you must install the newer database server in a different directory.*

Before you overwrite the older version, you must take the following precautions:

- If you do not have the original media for the older version, back up the **INFORMIXDIR** directory before you install your target database server.
- Copy the configuration file(s) in **INFORMIXDIR** in the **etc** directory to another location on the file system.

UNIX

When you finish the installation and system reconfiguration, exit as user **root** and log on as user **informix**. ♦

WIN NT

In Windows NT, you need to run the Installation wizard twice. First, upgrade the database server. Then install the administration tools.

The Installation wizard replaces the files but does not reconfigure the database server. If a previous version of the database server is on the computer, the **Upgrade** page appears when you install the new product.

The installation program automatically verifies and brings down your source database server, copies the new files, and preserves the database configuration information. The installation program starts your target database server with the same configuration and shared-server computer. ♦

Install Informix Enterprise Command Center

After you install the database server, install the administration tool, Informix Enterprise Command Center.

WIN NT

After you install the database server, IECC, or both, the Informix Administration Tools program group is available in your windowing environment. For complete information on installation, refer to your *Installation Guide for Informix Dynamic Server on Windows NT*. ♦

Customize the Database Server Environment

After you install your database server, ensure that the following environment variables are set to the correct values:

- **INFORMIXSERVER**
- **ONCONFIG**
- **PATH**
- **INFORMIXSQLHOSTS** (if used)

Important: *On UNIX, the client application looks for the* ***sqlhosts*** *file in the* ***etc*** *directory in the* ***INFORMIXDIR*** *directory. On Windows NT,* ***INFORMIXSQLHOSTS*** *points to the computer that contains the* ***sqlhosts*** *registry information. However, you can use the* ***INFORMIXSQLHOSTS*** *environment variable to change the location or name of the* ***sqlhosts*** *file.*

WIN NT

On Windows NT, the installation program sets the configuration parameters and environment variables for you. However, you can customize the configuration parameters in the **%ONCONFIG%** file and environment variables for your database server. In Windows NT, use **setnet32** to customize the environment variables on the client computer. ♦

For information about environment variables, refer to the *Informix Guide to SQL: Reference*.

UNIX

Update the ONCONFIG Configuration Parameters

You can customize your ONCONFIG configuration file and environment variables to take advantage of the new features introduced by Informix Dynamic Server, Version 7.3.

For example, you can add and adjust new ISM configuration parameters. After you observe the performance of your database server, you might want to make further adjustments.

Important: *Use the same values for ROOTOFFSET, ROOTSIZE, and ROOTPATH that you used for the earlier version of your database server. Also, use the same values for size and number of physical logs, logical logs, and for mirroring (if available).*

For information on how to configure, refer to your *Administrator's Guide*. For information about how to tune the configuration parameters, refer to your *Performance Guide*.

Update the Configuration Files

During the installation procedure, the install script checks the **etc** directory in the **INFORMIXDIR** directory for files named **config.arc**, **oper_deflt.arc, termcap, logevent.sh, sessalrm,** and **permalrm**. If these files do not exist, the install script provides them. If the files do exist, the install script does not overwrite the files. Instead, the install script provides additional files named **Config.arc**, **Oper_deflt.arc, Termcap, Logevent.sh, Sessalrm,** and **Permalrm** (note the initial uppercase letters).

Compare your current versions of the files with the new versions and determine whether new or changed configuration parameters or qualifiers exist.

Configure the Database Server to Use Enterprise Replication

Perform this step only if you plan to use Enterprise Replication or Workgroup Replication (Version 7.22 or later) with your database server.

Tip: *Informix Dynamic Server, Version 7.3, OnLine Workgroup Server, Version 7.2x, and OnLine Dynamic Server, Version 7.2x, on Windows NT only, can use Workgroup Replication.*

Before you can activate Enterprise Replication, you need to update the **$ONCONFIG** and **sqlhosts** files.

To activate Enterprise Replication for the first time

1. Bring your database server off-line.
2. Define one or more **dbserver aliases** for Enterprise Replication.
3. Define a **group name** for Enterprise Replication.
4. Bring your database server on-line.
5. For upgrades to versions 7.22 to 7.24, verify that the SNMP service is installed on the host and that the SNMP subagents are registered.

   In Informix Dynamic Server, Version 7.3, you need SNMP only if you plan to use the Enterprise Replication Monitoring Program.
6. Use Replication Manager to define each database server for replication.

   This step starts Enterprise Replication.

For complete information on how to configure for Enterprise Replication, refer to the *Guide to Informix Enterprise Replication*. For information on how to use SNMP, refer to the *Informix SNMP Subagent Guide*.

WIN NT

Installing and Configuring SNMP

In Informix Dynamic Server, Version 7.3, if you plan to use the Enterprise Replication Monitoring Program, you must install SNMP. Enterprise Replication in Informix Dynamic Server, Version 7.3 does not require SNMP. The SNMP management tools uses the **OnSnmpSubagent** to respond to queries. For Informix database servers, Version 7.22 through Version 7.24, if you use Workgroup Replication on Windows NT, the **onsnmp** utility requires Windows NT SNMP. The installation and upgrade program checks the registry for the SNMP master agent. If the master agent was not installed, the program displays a warning message but does not configure the registry for Workgroup Replication or **onsnmp**. If you later choose to install the SNMP master agent, you must run the **%INFORMIXDIR%\bin\inssnmp.exe** command line utility to install the SNMP subagents. You do not need to reinstall the database server.

Bring the Target Database Server On-Line

UNIX

When you bring your target database server on-line for the first time, bring it first to quiescent mode and then to on-line mode.

Execute the following command to bring your database server from off-line to quiescent mode:

```
oninit
```

Once the database server is in quiescent mode, check the message log for status messages.

Important: *If you note problems in the message file, solve the problems before you continue to the next step.*

Execute the following command to change your database server mode from quiescent mode to on-line mode:

```
onmode -m
```

The **sysmaster** and **sysutils** databases are created once your database server is brought on-line.

Warning: *The logical logs continue to fill with the transactions that result from the creation of the* ***sysmaster*** *database. If you run out of log space before the creation of the* ***sysmaster*** *database is complete, the database server halts with a "Logs Full" error. Thus, you must back up the logical logs.*

Use ON-Bar, ON-Archive, or **ontape** to back up logical logs. ♦

WIN NT

The setup program for Windows NT brings your database server on-line automatically.

If you customized the database server environment, bring down and restart the database server using the Informix Enterprise Command Center (IECC). When you restart your database server, the changes to the configuration parameters and environment variables take effect.

To start your database server with IECC

1. In the IECC Console program group, double-click the Informix Enterprise Command Center icon.
2. In the Informix Enterprise Command Center, select the database server in the tree view.
3. Choose **Server→On-line**.

You can start the database server with or without the Informix Enterprise Command Center interface. For more information, refer to *Informix Enterprise Command Center Installation Guide* or on-line help. ♦

Run UPDATE STATISTICS

After you complete the migration procedure, run the UPDATE STATISTICS statement according to the recommended procedure in the *Informix Guide to SQL: Syntax*. UPDATE STATISTICS updates the information that your database server uses to plan efficient queries. For more information about UPDATE STATISTICS, refer to the *Informix Guide to SQL: Syntax*.

Verify the Integrity of the Data

Before you allow users to access the databases, use the **oncheck** utility to verify that no data was corrupted in the migration process. You can verify the integrity of the reserve pages, system catalog tables, data, and indexes. For more information, see "Verify the Integrity of the Data" on page 5-22.

Check Space Requirements for BYTE and TEXT Data

If you are migrating from OnLine Dynamic Server 6.0 or 7.1, the space required for BYTE and TEXT data might have changed. Run **oncheck -cD** on the tables that have BYTE and TEXT data. This **oncheck** command modifies the internal bitmaps to show any changes in space availability. It also shows WARNING messages when a bitmap has been changed. For more information, refer to "Changes to Blobspace Requirements in Version 7.10.UD1" on page 5-16.

Make an Initial Backup

Use your database server backup utility (ON-Bar, ON-Archive, or **ontape)** to make a level-0 backup. Do not overwrite the tapes you used earlier when you made your final backup of your database sever. For more information about making a backup, refer to the *Archive and Backup Guide for Informix Dynamic Server* or the *Backup and Restore Guide for Informix Dynamic Server*.

Important: *Do not restore the backed-up logical logs from the earlier version of your database server to the newer version of your database server.*

Verify the Access Path of Your SQL Statements

Use the SET EXPLAIN statement to verify that the access path of your SQL statements did not change when you migrated to your target database server. If you have SET EXPLAIN output from the source database server, run SET EXPLAIN for your target database server. Compare the SET EXPLAIN output from both the source and target database servers. SET EXPLAIN writes the access path that the optimizer chooses for each query to the SET EXPLAIN output file. The optimizer chooses the fastest path of execution for table joins.

WIN NT

For Windows NT, the SET EXPLAIN output filename is **%INFORMIXDIR%\sqlexpln\*username*.out**. ♦

If the SET EXPLAIN output file shows that a different access path was used, complete the following steps:

1. Check the **OPTCOMPIND** environment variable or configuration parameter.
2. Check the **DBSPACETEMP** environment variable or configuration parameter to ensure that adequate temporary dbspaces are defined. You might need to define more temporary dbspaces.
3. Analyze the query access paths, and modify the schema to improve the performance if necessary.

Migration Complete

The first time your database server is brought on-line, the **sysmaster** database is built. Check the message log to ensure that the **sysmaster** database build is complete before you allow users to access the database server. After you complete a level-0 backup and you ensure that client users can access data on your database server, the migration process is complete.

Once you successfully migrate to Informix Dynamic Server, Version 7.3, you might want to seek ways to obtain maximum performance. If you created sample queries for comparison, you can use them to characterize the performance differences between your earlier version and your new version. The results of these comparisons might suggest adjustments to configuration parameters, or to the layout of databases, tables, and chunks. For details on topics related to performance, refer to the *Performance Guide for Informix Dynamic Server*.

WIN NT

Changing Database Server Definitions

The installation program automatically updates the registry information.

Use Informix Enterprise Command Center if you want to change the **informix** user password, specify a different computer as the shared-server computer, or edit database server definitions (the database server name, TCP/IP host names, network types, and the service names). For example, you might specify a different shared-server computer if you have migrated the database server to a new computer or connected a single client to several database servers.

For information on how to use the **Client Control Panel** folder in Informix Enterprise Command Center to configure database servers, see *Informix Enterprise Command Center User Guide* or the on-line help. For information on connectivity, see your *Administrator's Guide*.

GLS

Migrating to a GLS Locale

If you wish to migrate your database server to a non-English GLS locale, set the DB_LOCALE and CLIENT_LOCALE environment variables before you open the database in Version 7.3. If your previous database server version used Native Language Support (NLS), replace the NLS environment variables, such as **COLLCHAR**, with GLS environment variables. For information on how to work with locales and how to set GLS environment variables, see the *Informix Guide to GLS Functionality* and Chapter 10, "Changing Locales."

Reverting to OnLine Dynamic Server

This section covers the changes that occur when you revert to an earlier version of the database server. You can revert between the following versions of the database server.

Revert From	Revert To
Informix Dynamic Server Version 7.3	OnLine Dynamic Server Version 7.2x
OnLine Dynamic Server Version 7.24	Previous OnLine Dynamic Server Version 7.2x
OnLine Dynamic Server 7.22	OnLine Dynamic Server 7.2
OnLine Dynamic Server 7.2x	OnLine Dynamic Server 7.10.UD1 through 7.14
OnLine Dynamic Server 7.2x	OnLine Dynamic Server 7.1
OnLine Dynamic Server 7.2x	OnLine Dynamic Server 6.0
OnLine Dynamic Server 7.2x	OnLine Dynamic Server 5.0
OnLine Dynamic Server 7.2x	OnLine Dynamic Server 4.1
OnLine Dynamic Server 7.1x (later version)	OnLine Dynamic Server 7.1x (earlier version)

Reversion Steps

This section describes the steps to revert to an earlier version of your database server. When you revert, you must consider changes in the definitions of configuration parameters and environment variables.

Follow the preparatory steps, described in "Preparing to Migrate Between Versions" on page 5-18, then complete the following steps:

1. Remove unsupported SQL features.
2. Stop and uninstall Enterprise or Workgroup Replication, if it is installed.
3. Uninstall the database server and the administration tools.
4. Modify In-Place ALTER TABLE (Version 7.24 and later).
5. Close all transactions and bring the database into quiescent mode.
6. Verify the integrity of the data.
7. Back up your database server.
8. Run the reversion utility (**onmode -b**).
9. Modify configuration parameters.
10. Reset environment variables.

WIN NT

11. Start the desired version of your database server. ♦
12. Reinstall the old version of the database server.
13. Start the database server.
14. Verify the integrity of the data.
15. Back up the database server files.
16. Return the database server to on-line mode.

Warning: *On Windows NT, you need to uninstall Enterprise Replication Manager and your database server and then reinstall the old version of your database sever. You cannot have two versions installed concurrently.*

If you revert to Version 7.1 or Version 6.0, you also need to perform the following steps before your start the desired version:

1. Remove fragmented tables (only when you are reverting to Version 6.0).
2. Save system catalog information.
3. Remove GLS features.
4. Modify the **sqlhosts** file.
5. Rename ON-Archive files (only for Version 6.0).

For Version 6.0, after you install the database server, restore system catalog information.

Remove Unsupported SQL Features

Before you revert, you must remove SQL features that the earlier version of your database server does not support. See the "New Features of This Product" section in the appropriate version of the *Informix Guide to SQL: Syntax*.

UNIX

Remove Fragmented Tables (Version 6.0 Only)

If you are reverting to Version 6.0, you must change all fragmented tables back into unfragmented tables. For information on how to defragment tables, see your *Administrator's Guide*.

Save System Catalog Information

If your current database server instance uses secure-auditing masks or ON-Archive, and you want to preserve the associated catalog information, you must unload these system catalog tables before you continue. Execute the following command to unload the system catalog tables:

```
$INFORMIXDIR/etc/smi_unld
```

When the **smi_unld** utility finishes unloading the information, the utility displays instructions for reloading the information. *Save these instructions.*

After you complete the reversion and initialize your database server, you can reload the data that you preserved. Follow the instructions given with the **smi_unld** utility for reloading the information. Typically, you should execute the following command:

```
$INFORMIXDIR/etc/smi_load $INFORMIXDIR/etc/
```

Stop Enterprise or Workgroup Replication (Version 7.22 or Later)

Skip this section if Enterprise or Workgroup Replication is not installed on your system.

To revert to an earlier version if Enterprise or Workgroup Replication is active

1. Stop Enterprise or Workgroup Replication.

UNIX

2. For altered tables with CRCOLS, issue the command:

   ```
   alter table drop CRCOLS
   ```

   ♦

3. Execute the **onmode -b** command to revert to the earlier version of your database server.

Warning: *If you try to revert to a previous version of the database server while Enterprise Replication is active, the reversion will fail.*

To revert to an earlier version if Enterprise or Workgroup Replication is inactive

UNIX

1. In this situation, Enterprise Replication was previously active on this database server. For altered tables with CRCOLS, issue the command:

   ```
   alter table drop CRCOLS
   ```

   ♦

2. Execute the **onmode -b** command to revert to the earlier version of the database server. The **syscdr** database is dropped during reversion.

For more information, see the *Guide to Informix Enterprise Replication*.

Uninstall the Database Server and INFORMIX-Enterprise Command Center

Uninstall both the database server and the administration tool, Informix Enterprise Command Center.

WIN NT

To uninstall the product

1. Double-click the **Uninstall** icon in the IECC Console program group.
2. In the Uninstall dialog box, check **Remove OnLine Server**.

Warning: *Do not check **Remove all OnLine databases, supporting files and all database information**. If you check this option, your configuration, dbspaces, and database information will be lost, making reversion impossible.*

3. To uninstall the administration tools, check **Remove Administration Tools**.
4. Click **OK** to uninstall the database server.

For more information, see the *Informix Enterprise Command Center Installation Guide*.

Modify In-Place ALTER TABLE (Version 7.24 and Later)

Upgrading to the database server, Version 7.24, and later occurs automatically. However, reverting from Version 7.24 is not possible if outstanding In-Place ALTER TABLEs exist. An In-Place ALTER TABLE is outstanding when data pages exist with the old definition.

If you attempt to revert to a previous version, the code checks for outstanding alter operations and lists any that it finds. You need to update every row of each table in the outstanding alter list with an ALTER TABLE version and then perform the reversion.

If an In-Place ALTER TABLE was performed on a table, you can convert the older version pages to the latest version by running a test UPDATE statement.

For example, run the following test UPDATE statement:

```
update tab1 set column1 = column1
```

For more information on In-Place ALTER TABLE, see your *Performance Guide*.

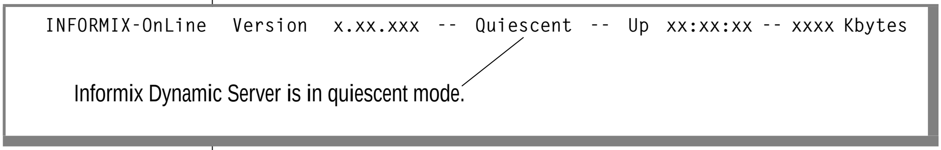
Close All Transactions and Bring the Database into Quiescent Mode

Communicate to client users how long you expect the database server to be off-line for the migration. Terminate all database server processes, and place your database server in quiescent mode (also called *administration mode*).

To shut down the database server gracefully

1. Warn all users that you plan to shut down the database server and wait for them to exit.
2. Become user **informix** on UNIX platforms, or on Windows NT, you must be a member of the **Informix-Admin** group.
3. Execute the **onmode –sy** command.
4. To verify the mode of your database server, execute the **onstat -** command.

   The first line of the **onstat** output contains the status of your database server. Figure 5-6 shows that the database server is in quiescent mode.



```
INFORMIX-OnLine  Version  x.xx.xxx  --  Quiescent  --  Up  xx:xx:xx -- xxxx Kbytes
```

Informix Dynamic Server is in quiescent mode.

Figure 5-6
Example of onstat Status Line

5. Execute the following command to force a new logical log:

   ```
   onmode -l
   ```

6. Execute the following command to force a checkpoint:

   ```
   onmode-c
   ```

Important: *Monitor your log to verify that all commands were executed properly and to check for inconsistencies prior to migration.*

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of data before you make a level-0 (complete) backup. If you find any problems with the data, fix them before you make the backup. You can verify the integrity of the reserve pages, extents, system catalog tables, data, and indexes. First, you need to obtain a list of the databases on your database server.

Figure 5-7 lists the **oncheck** commands that verify data integrity. Informix recommends that at a minimum you run the **-cr** and **-ce** options.

Figure 5-7 *Commands for Verifying the Data Integrity*

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *`database_name`*
Check data	`oncheck -cD` *`database_name`*
Check indexes	`oncheck -cI` *`database_name`*

For information on **oncheck**, refer to your *Administrator's Guide*.

Back Up Your Database Server

Informix recommends that you make a level-0 backup of each database server that you plan to migrate. Use ON-Bar, ON-Archive, or the **ontape** utility to perform the backup.

WIN NT

The Windows NT environment does not support ON-Archive. ♦

For information about how to perform a backup, refer to the *Archive and Backup Guide for Informix Dynamic Server*.

Warning: *Backups that you perform under older versions of your database server are not compatible with the newer version. Do not try to restore these backups to the newer version.*

Run the Reversion Utility

The database server must be running when you execute the reversion utility. You must use this reversion utility to restore compatibility before users can access the data with the earlier version. The **onmode** utility does not revert changes made to the layout of the data that do not affect compatibility.

To revert to an earlier version of the database server

1. Execute the reversion utility to revert the database, where *version_number* is the earlier database server version (see Figure 5-8):

   ```
   onmode -b version_number
   ```

2. Drop the **sysutils** database if the earlier version does not support ON-Bar.

The reversion utility forcibly removes all users and shuts down the database server. After the reversion is complete, the database server ID is off-line. For more information about the **onmode -b** command, refer to "The onmode Utility" on page 11-69.

Figure 5-8
Reverting to an Earlier Version of the Database Server

Revert From	Revert To	Command
Version 7.3 / NT / UNIX	Version 7.22 to Version 7.24 / NT / UNIX	**onmode -b 7.24**, **onmode -b 7.23**, or **onmode -b 7.22**
Version 7.24 / NT / UNIX	Version 7.22 / NT / UNIX	Reversion is automatic
Version 7.22 / NT / UNIX	Version 7.20.UC1 and prior to 7.30UC1 / NT / UNIX	**onmode -b 7.2**
Version 7.2 through 7.22 / UNIX	Version 7.10.UD1 and prior to 7.20.UC1 / UNIX	**onmode -b 7.10.UD1**
Version 7.2 through 7.22 / UNIX	Version 7.10.UC1 and prior to 7.10.UD1 / UNIX	**onmode -b 7.1**
Version 7.10.UD1 through 7.14 / UNIX	Version 7.10.UC1 and prior to 7.10.UD1 / UNIX	**onmode -b 7.1**

(1 of 2)

Revert From	Revert To	Command
Version 7.2 through 7.22 / UNIX	Version 6.0xx except Version 6.0ALS / UNIX	**onmode -b 6.0**
Version 7.2 through 7.22 / UNIX	Version 6.0 / UNIX / ALS	**onmode -b 6.0A**
Version 7.12 through 7.14 / UNIX	Version 7.10.UD1 / UNIX	Reversion is automatic
Version 6.0 / UNIX	The latest maintenance Version 5.0x / UNIX	**onmode -b 5.0**

(2 of 2)

***Tip:** The **onmode -b** command also rebuilds the user-table indexes automatically.*

Remove GLS Features

Skip this step if your database server uses the default English locale (**en_us.8859-1**). To revert the database server from GLS to Native Language Support (NLS) or Asian Language Support (ALS), set the appropriate NLS or ALS locales and environment variables. For information on working with locales, see the *Informix Guide to GLS Functionality* and Chapter 10, "Changing Locales."

Modify Configuration Parameters

Informix Dynamic Server, Version 7.3 uses configuration parameters that did not appear in earlier versions of the database server. You may want to remove or modify these configuration parameters. For more information, see "Configuration Parameters Introduced in Version 7.3" on page 5-6.

Configuration Changes for Version 7.22

Version 7.22 changed the default value of the ALARMPROGRAM configuration parameter. You might need to revise the values in your configuration file.

Remove the following parameters from your ONCONFIG file:

- CDR_LOGBUFFERS
- CDR_EVALTHREADS
- CDR_DSLOCKWAIT
- CDR_QUEUEMEM
- CDR_NIFUSEHELP
- CDR_NIFMEMS
- SCDR_NIFQUEUES

WIN NT

Configuration Changes for Version 7.12

You must add the ONLANGMAP configuration parameter to your ONCONFIG file before you start OnLine Dynamic Server, Version 7.12. ONLANGMAP takes the value **en_US-English**. For information on ONLANGMAP, see the documentation notes for INFORMIX-OnLine Dynamic Server for Windows NT 3.51, Version 7.12.TC2.

If you are reverting to Version 7.12, remove the HETERO_COMMIT configuration parameter from your ONCONFIG file.

Version 7.22 changed the default value of the ALARMPROGRAM configuration parameter. You might need to revise the values in your ONCONFIG file.

Configuration Changes for Version 7.10.UD1 Through 7.14

If you are reverting to Version 7.10.UD1 through Version 7.14, remove the following parameters from your ONCONFIG configuration file:

- CHUNKS
- DBSPACES
- HETERO_COMMIT
- TBLSPACES
- TRANSACTIONS
- USERTHREADS

Version 7.2x changed the definition of the following two configuration parameters. You might need to revise the values in your configuration file:

- BUFFERS
- MAX_PDQPRIORITY

Version 7.2x removed the following configuration parameters. If you do not restore the parameters to your ONCONFIG file, OnLine Dynamic Server uses the default values. You can add these parameters to your ONCONFIG file:

- BUFFSIZE
- PDQPRIORITY

Configuration Changes for Version 7.10.UCx

If you are reverting to Version 7.10.UC1, remove the following parameters from your ONCONFIG configuration file:

- LBUPRESERVE
- ONDBSPACEDOWN
- OPCACHEMAX

Version 7.10.UD1 or 7.2 changed the definition of the following configuration parameters. You might need to revise the values in your configuration file:

- BUFFERS (changed in 7.10.UD1)
- LOCKS (changed in 7.2)
- MAX_PDQPRIORITY (changed in 7.10.UD1)
- OPTCOMPIND (changed in 7.2)

Version 7.10.UD1 removed the following configuration parameters. If you do not restore the parameters to your ONCONFIG file, OnLine Dynamic Server uses the default values. You should restore these parameters to your ONCONFIG file:

- CHUNKS
- DBSPACES
- PDQPRIORITY (removed in 7.2)
- TBLSPACES
- TRANSACTIONS
- USERTHREADS

Version 7.10.UD1 moved the following audit configuration parameters into the audit configuration file (**$INFORMIXDIR/aaodir/adtcfg.std**). If you use the ON-Audit utility, restore these parameters to your ONCONFIG file:

- ADTPATH (UNIX only)
- ADTSIZE (UNIX only)
- ADTERR
- ADTMODE

Configuration Changes for Version 6.0

If you are reverting to Version 6.0, make the changes specified in "Configuration Changes for Version 7.12" and "Configuration Changes for Version 7.10.UCx" on page 5-43. In addition, remove the following parameters from your ONCONFIG configuration file:

- ALARMPROGRAM
- DATASKIP
- DS_MAX_QUERIES
- DS_MAX_SCANS
- DS_TOTAL_MEMORY
- MAX_PDQPRIORITY
- OPTCOMPIND
- PDQPRIORITY

Important: *Use the same values for ROOTPATH, ROOTSIZE, and ROOTOFFSET in your ONCONFIG configuration file for both versions of your database server.*

Reset Environment Variables

Reset the environment variables to values that are appropriate for your version of your database server. For information on environment variables, refer to the *Informix Guide to SQL: Reference*.

NLS

Environment Variable Changes for Version 7.12

OnLine Dynamic Server, Version 7.12, supports NLS, not GLS. When you revert to Version 7.12, delete the following environment variables:

- **CC8BITLEVEL**
- **CLIENT_LOCALE**
- **DBCENTURY**
- **DBFLTMASK**
- **DBONPLOAD**
- **DB_LOCALE**
- **ESQLMF**
- **GLS8BITFSYS**
- **GL_DATES**
- **GL_DATETIME**
- **ONPLOAD**
- **PLCONFIG**
- **SERVER_LOCALE**
- **THREADLIB**

Add the following environment variables:

- **DBNLS**
- **COLLCHAR**
- **LANG**

NLS

Environment Variable Changes for Version 7.10.UD1 through 7.14

OnLine Dynamic Server, Version 7.10.UD1 through 7.14, supports NLS, not GLS. When you revert, delete the following environment variables:

- **CC8BITLEVEL**
- **CLIENT_LOCALE**
- **DBCENTURY**
- **DBONPLOAD**
- **DB_LOCALE**
- **ESQLMF**
- **GLS8BITSYS**
- **GL_DATE**
- **GL_DATETIME**
- **NODEFDAC**
- **ONPLOAD**
- **PLCONFIG**
- **SERVER_LOCALE**
- **THREADLIB**

Environment Variable Changes for Version 7.10.UCx

When you revert to Version 7.10.UCx, do not use the following environment variables:

- **INFORMIXOPCACHE**
- **INFORMIXSQLHOSTS**
- **NODEFDAC**

The recommended settings of the following environment variables changed between Version 7.1 and Version 7.2x. You might need to reset their values:

- **OPTCOMPIND**
- **PSORT_NPROCS**

Environment Variable Changes for Version 6.0

When you revert to Version 6.0, make the environment variable changes that are specified for Version 7.1. In addition, do not use the following environment variables:

- **DELIMIDENT**
- **FET_BUF_SIZE**
- **PDQPRIORITY**

The **PDQPRIORITY** environment variable and the SQL SETPDQPRIORITY statement were introduced after Version 6.0. The **PDQPRIORITY** environment variable does not cause problems if you leave it set for Version 6.0, but it might cause confusion. You must remove the SETPDQPRIORITY statement from your Version 6.0 applications.

UNIX

Modify the sqlhosts File

Version 7.10.UD1 introduced the following enhancements to the **sqlhosts** file. If you use any of these enhancements, you must modify your **sqlhosts** file before you run Version 7.10.UCx or Version 6.0:

- Host-name length of 256 characters
- **INFORMIXSQLHOSTS** environment variable
- Stream pipes
- Explicit addressing for TCP/IP
- Options field

WIN NT

Reinstall the Earlier Version of the Database Server

Reinstall the earlier version of the database server in the same directory as the files for the previous version.

To perform the installation

1. On the **Run Installation Again** page, select the **Copy all files, but leave configuration alone** installation option.
2. Supply your serial number and serial number key, as shown on the serial-number key card.
3. Select one or both of the components you want to install: **OnLine Database Server** or **Administration Tools**. Click **Next**.
4. The installation program automatically copies the older database server files and saves the configuration and the database data.

For more information on installation, refer to the *INFORMIX-Enterprise Command Center User Guide*.

Warning: *Do not select* ***Copy all files and reconfigure the product****. If you select this option, your configuration and database information will be lost.*

Bring the Database Server On-Line

Execute the following command to bring your database server to quiescent mode:

```
oninit
```

The database server initializes the shared memory and builds the **sysmaster** database. After the **sysmaster** database is built, the reversion process is complete.

Important: *You must add the ONLANGMAP parameter to the ONCONFIG file before you start OnLine Dynamic Server 7.12 or earlier. ONLANGMAP takes the value **en_US-English**. The ONLANGMAP configuration parameter ensures backward compatibility with non-NLS locales on Windows NT. If the database server fails on install, just restart it.*

WIN NT

To start the database server (for Version 7.12 only)

1. From the Windows NT **Main** program group, double-click the **Control Panel** icon.
2. Double-click the **Services** icon.
3. Select **INFORMIX**-OnLine Dynamic Server from the **Services** list box.
4. Click **Start**.

   The Services dialog box displays the status of the database server.

If you revert to Version 7.22 on Windows NT, you can start the database server from INFORMIX-Enterprise Command Center.

Warning: *If you start the database server with the **-iy** parameters after the first time, it will overwrite the existing root dbspace unless you first change the ROOTPATH parameter in the ONCONFIG file. If the database server overwrites the existing root dbspace, it destroys the information that the root dbspace contains, including information about any databases that you created. Consequently, you must then restore the databases from backup tapes.*

Rename ON-Archive Files (Version 6.0 Only)

The names of the ON-Archive error message and help files changed between Version 6.0 and Version 7.1. For the correct names for these files, refer to the *INFORMIX-OnLine Dynamic Server Archive and Backup Guide*, Version 6.0.

Restore System Catalog Information (Version 6.0 Only)

After you initialize your database server, you might need to restore system catalog information. Follow the instructions in "Save System Catalog Information" on page 5-35.

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of data before you make a level-0 (complete) backup. If you find any problems with the data, fix them before you make the backup. You can verify the integrity of the reserve pages, extents, system catalog tables, data, and indexes. First, you need to obtain a list of the databases on your database server.

Figure 5-5 lists the **oncheck** commands that verify data integrity. Informix recommends that at a minimum you run the **-cr** and **-ce** options.

Figure 5-9 *Commands for Verifying the Data Integrity*

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *`database_name`*
Check data	`oncheck -cD` *`database_name`*
Check indexes	`oncheck -cI` *`database_name`*

For information on **oncheck**, refer to the your *Administrator's Guide*.

Back Up the Database Server Files

After you complete the reversion, Informix recommends that you make a level-0 backup. Use your preferred backup administration utility, ON-Bar, ON-Archive, or **ontape**, to make the backup. For information about how to make a backup, refer to your *Archive and Backup Guide*.

Important: *Do not overwrite the tapes that you used to back up your database server.*

Return the Database Server to On-Line Mode

To bring the database server on-line, execute the following command:

```
onmode -m
```

Reversion Is Now Complete

Ensure that client users can access data on the earlier version of the database server.

Reverting from Version 7.10.UDI, 7.12, and 7.14 to 7.10.UCx

OnLine Dynamic Server 7.10.UCx is not compatible with later 7.1x versions of the product. For example, if you wish to revert from Version 7.12 to Version 7.10.UCx, you must run the **onmode -b** command to restore the data to a form that is compatible with the earlier version.

OnLine Dynamic Server versions 7.10.UD1, 7.11 through 7.14 are compatible with each other, so you do not use **onmode -b** when you migrate the data.

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of data before you make a level-0 (complete) backup. If you find any problems with the data, fix them before you make the backup. You can verify the integrity of the reserve pages, extents, system catalog tables, data, and indexes. First, you need to obtain a list of the databases on your database server.

Figure 5-10 lists the **oncheck** commands that verify data integrity. Informix recommends that at a minimum you run the **-cr** and **-ce** options.

Figure 5-10
Commands for Verifying the Data Integrity

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *`database_name`*
Check data	`oncheck -cD` *`database_name`*
Check indexes	`oncheck -cI` *`database_name`*

For information on **oncheck**, refer to the your *Administrator's Guide*.

Back Up the Database Server

After you complete the reversion, Informix recommends that you make a complete backup. Use the **ontape** utility to make the backup. For information about how to use **ontape**, refer to your *Archive and Backup Guide*.

Important: *Do not overwrite the tapes that you used to back up your database server.*

Reversion Is Now Complete

Ensure that client users can access data on the earlier version of your database server.

Chapter

6

Migrating Pre-6.0 to Informix Dynamic Server 7.3

This chapter describes how to upgrade from pre-6.0 versions of INFORMIX-OnLine to Informix Dynamic Server, Version 7.3. You can use the same procedure to upgrade to versions 7.2, 7.21, 7.22, 7.23, or 7.24. For new features in Informix Dynamic Server, Version 7.3, refer also to Chapter 5, "Migrating the Database Server 6.0 or Later to 7.3."

Preparing to Migrate Between Versions

When you migrate from one version of any database server to another, Informix suggests that you follow these guidelines:

- Review the release notes for the version of your database server for information about new features, installation, and fixes to problems. Modify applications as needed.

  The release notes are in one of the following directories:

  UNIX

  ❑ **$INFORMIXDIR/release/en_us/0333**. ♦

  WIN NT

  ❑ **%INFORMIXDIR%\release\en_us\0333 (**as of Version 7.2)

  Release Notes appear in the Informix folder. To display this folder, choose **Start→Programs→Informix** from the Task Bar. ♦

- On UNIX, retain both versions of the Informix product software on disk (if you have enough disk resources). On Windows NT, you cannot retain two versions of the Informix product on disk.
- Check the documentation notes for information about features not covered in the manuals.

- Retain the installation tapes from both versions of the Informix product software.
- Make a level-0 backup of the database server before and after migration.

In addition to the material in this chapter, read the following information in "Changes Introduced in Database Server Version 6.0 to Version 7.3" on page 5-6.

For installation and configuration, refer also to both your *Installation Guide* and your *Administrator's Guide*.

Tip: *This chapter does not discuss migration for Windows NT. For information on migration for Windows NT, see Chapter 9, "Moving Between Database Servers."*

Changes Introduced in Version 6.0

In Version 6.0, OnLine Dynamic Server introduced an architecture that differs greatly from the architecture used in OnLine Dynamic Server 4.1 or 5.0. This new architecture requires significant changes in allocation and disk use. Figure 6-11 describes the major changes.

Figure 6-1
Changes Introduced in Version 6.0

Area of Change	Comments
Backup-tape format	Backups and logical-log backups made with pre-6.0 **tbtape** are not compatible with either of the two Version 7.2x tape utilities, ON-Archive and **ontape**. Informix recommends that you make a backup before you convert to Version 7.2x, and then make a second backup once the conversion is complete.
sqlhosts file	The **sqlhosts** file is mandatory. You must create an **sqlhosts** file or modify your current **sqlhosts** file to the format for OnLine Dynamic Server 7.2x.
Utility names	Version 6.0 introduced a new naming convention for the utilities. The INFORMIX-OnLine Dynamic Server utilities use the prefix **on** (for example, **oninit**) and the INFORMIX-SE utilities use the prefix **se** (for example, **selog**).

(1 of 2)

Area of Change	Comments
sysmaster database	When you initialize Informix Dynamic Server, Version 7.3, a script automatically creates the **sysmaster** and **sysutils** databases. You must ensure that at least 1,100 free pages exist in the root dbspace to build this database.
System resources	The Version 6.0 changes include new requirements for system resources such as shared memory, semaphores, and disk space. When you migrate from OnLine Dynamic Server 4.1 or 5.0 to a later version, you must reconfigure the operating-system kernel.
Index requirements	To accommodate new features such as key-value locking, the indexing scheme requires an additional 1 byte of disk space per index-key entry. You must rebuild all user indexes after you migrate from OnLine Dynamic Server 4.1 or 5.0. An index on a table with one million records requires approximately 1 additional megabyte of disk space.

(2 of 2)

Changes to Database Utilities

Utility names that began with **tb** in pre-6.0 versions begin with **on** in OnLine Dynamic Server 6.0 and later. For instance, **tbcheck** was replaced by **oncheck**. For the complete list of utilities, refer to your *Administrator's Guide*. Plan to update all references to **tb*** utilities.

You cannot use utilities that use binary formats to load and unload data between post-6.0 and pre-6.0 versions. To transfer data between post 6.0 versions and pre-6.0 versions, you can use only those utilities that load and unload data in ASCII format. For instance, the **tbload**, **tbunload**, and **tbtape** utilities in Version 4.1 and 5.0 use binary data and you cannot use them to transfer data to a Version 7.3 database server. The **onload**, **onunload**, and **ontape** utilities in Version 7.3 use binary data and you cannot use them to transfer data to earlier versions. You can use **dbexport** (pre-6.0 version) and **dbimport** (Version 7.3) to transfer ASCII data to a Version 7.3 database server. You can also use the SQL LOAD and UNLOAD statements to transfer data between versions.

Planning for Migration

This section describes the planning and preparations requirements to migrate to database servers after Version 6.0, because of the changes that occurred between Version 5.0 and Version 6.0.

Operating-System Configuration Issues

Database servers later than Version 6.0 require system resources in addition to those required in OnLine Dynamic Server 4.1 or 5.0. These resources include additional shared-memory segments, additional semaphores, and additional open-file descriptors per process.

The specific tunable parameters and methods that you use to configure these resources into the operating system vary from platform to platform. For more information, consult the machine-notes file that is installed with your distribution of your target database server and the configuration instructions for your operating system.

Estimating the Size and Number of Shared-Memory Segments

When you move from Version 4.1 or 5.0, you must recalculate your memory requirements. Database servers later than Version 6.0 make more extensive use of shared memory than OnLine Dynamic Server 4.1 or 5.0. For example, in database servers later than Version 6.0, in addition to housing the buffer cache, virtual processors use shared memory to manage user threads and other activities that individual server processes handled in earlier versions. When you upgrade from OnLine Dynamic Server 4.1 or 5.0 to Version 7.2x or later, add an additional 8 megabytes of shared memory.

From the standpoint of the operating system, virtual memory that previously was allocated to individual server processes in Version 4.1 and 5.0 is now included in the virtual segment attached by Version 7.2x or later.

This new arrangement requires a new method for calculating shared-memory requirements in the target database server, that takes the following items into account:

- The *virtual segment*, which is used to manage multiple user threads, data distributions, and other data
- The familiar RSAM or *resident segment*, that is used to manage the buffer cache
- A new *message segment*, which is used to support the shared-memory communication interface

Tip: *Data distributions in the database server provide the query optimizer with statistical information about the contents of columns and tables. For information about data distributions, refer to UPDATE STATISTICS in the "Informix Guide to SQL: Syntax."*

The overall system requirement for user virtual memory includes all three shared-memory segments as well as the space that is needed to hold process images. However, the system requirement for actual physical memory includes only the resident segment and the working sets from other segments and processes. Thus, the physical memory that the target database server requires is proportional to the resident segment, while the requirement for swap space is proportional to the total amount of shared memory that the database server uses.

AD/XP
IUS
UD
UNIX

After Version 6.0, Informix Dynamic Server, OnLine Dynamic Server, and Universal Server for database servers on UNIX require at least one virtual shared-memory segment. The default size of a shared-memory segment is 8 megabytes. ♦

You can use the following steps to generate a rough estimate for the size and number of shared-memory segments that are required for your instance of your target database server:

1. Estimate the total amount of shared memory that you need to initialize the target database server. You must make separate estimates for each of the three shared-memory segments, as described in the following paragraphs, and add up the total:

   - For an initial estimate of the resident segment size, use the size of shared memory as displayed in the output of **tbmonitor** under OnLine Dynamic Server 4.1 or 5.0. Because the resident segment in your target database server does not include big buffers, you can deduct 4 kilobytes for each 100 buffers in the BUFFERS parameter of your OnLine Dynamic Server 4.1 or 5.0 **tbconfig** file.
   - Depending on your application, an initial estimate for the virtual segment might be as low as 100 kilobytes per user, or as high as 500 kilobytes per user, plus up to 4 megabytes in addition if you intend to use data distributions. You can obtain an estimate of the number of users under your target database server by adding 12 to the value of the USERS parameter in your OnLine Dynamic Server 4.1 or 5.0 **tbconfig** file. The initial size of the virtual segment corresponds to the SHMVIRTSIZE configuration parameter in your target database server configuration file.
   - Use the following formula to estimate the size of the message segment:

     ```
     msgseg = (10,531 * connections) + 50,000
     ```

     connections is the number of user sessions that can connect through the shared-memory interface. You can set the number of sessions with the NETTYPE parameter in the OnLine Dynamic Server 7.2x ONCONFIG file.

   After you start your target database server, you can obtain a more precise value for SHMVIRTSIZE with **onstat –g mem**. You can then reconfigure shared memory more precisely with the actual value for SHMVIRTSIZE reported by this command.

2. The database server, Version 7.3, can attach additional shared-memory resources during operation when it performs a large sort or other operation that might require more memory than it has previously acquired. To allow the database server to expand its use of shared memory while it is operating, reserve a suitable margin of shared memory over that which you estimate is necessary to initialize your target database server. The SHMADD parameter in the ONCONFIG file specifies the size of a dynamically added segment. If you do not specify a value for this parameter in your ONCONFIG file, the database server attempts to attach additional shared memory in 8-megabyte segments.

   The SHMTOTAL parameter in the ONCONFIG file places an absolute maximum on the amount of shared memory that an instance of your target database server can request. To avoid the risk of exceeding the shared memory provided for a given instance of your target database server, you can set this parameter to indicate the maximum amount of shared memory for that instance. If you set SHMTOTAL to 0 or leave it unassigned, the database server continues to attach additional shared memory as needed until no more virtual memory is available on the system.

UNIX

3. Estimate the size and number of shared-memory segments that the operating system needs to provide, and then modify your kernel.

   If your operating system *does not* have a segment-size limit, take the following actions:

   a. Set the maximum-segment-size parameter, typically SHMMAX or SHMSIZE, to the total size that is required for your target database server. Include both the amount of memory that is required to initialize the database server as calculated in step 1 on page 6-8, and the amount of memory that you wish to allow for dynamic growth as described in step 2.

   b. Set the operating-system configuration parameter for the maximum number of segments, typically SHMMNI, to at least 1 per instance of the database server.

If your system *does* have a segment-size limit, take these actions:

a. Set the maximum segment-size parameter for the operating system, typically SHMMAX or SHMSIZE, to the largest value that your system allows.

b. Use the following formula to calculate the number of segments for your instance of your target database server:

```
SHMMNI =((initial_segment) / SHMMAX) + dynamic_segments
```

initial_segment is the segment size that is required to initialize your target database server.

dynamic_segments is the number of segments that you allow to be added during operation of your target database server.

If a remainder exists, round up to the nearest integer value.

4. If your operating system uses the SHMSEG configuration parameter to indicate the maximum number of shared-memory segments to which a process can attach, set this system-configuration parameter to a value that is equal to or greater than the largest number of segments that you allocated for any one instance of your target database server. ♦

UNIX

Configuring Semaphore Parameters

The operating-system configuration parameters for semaphores are calculated differently for OnLine Dynamic Server 7.1 or later than for pre-6.0 versions of the database server. On UNIX systems, the SEMMNI parameter gives the number of semaphore sets. Each instance of your target database server requires one set of semaphores for each group of (up to) 100 virtual processors (VPs) that are initialized with the database server, one set for each additional VP that you might add dynamically (while the database server is running), and one set for each group of 100 (or fewer) user sessions that are connected through the shared-memory communication interface. Because the target database server utilities such as **onmode** use shared-memory connections, you must configure a minimum of two semaphore sets for each instance of your target database server: one for the initial set of VPs and one for the shared-memory connections that the database server utilities use.

The SEMMSL operating-system configuration parameter typically gives the maximum number of semaphores per set; set this parameter to (no less than) 100.

On systems that require you to configure a maximum for the total number of semaphores across all sets, typically given by the SEMMNS operating-system configuration parameter, use the following formula to calculate the total required for each instance of your target database server:

```
SEMMNS = init_vps + added_vps + shmem_users + concurrent_utils
```

init_vps is the number of VPs that are initialized with your target database server. This number includes CPU, PIO, LIO, AIO, SHM, TLI, SOC, and ADM VPs. (For a description of these virtual processors, see your *Administrator's Guide*.) The minimum value for this term is 15.

added_vps is the number of VPs that you can add dynamically.

shmem_users is the number of shared-memory connections that are allowed for this instance of your target database server.

concurrent_utils is the number of concurrent database server utilities that can connect to this instance. Informix suggests that you allow for a minimum of six utility connections: two for **onarchive** and four for other utilities such as **onmonitor, oncheck**, and **onstat**.

For example, if you start a single instance of your target database server with two CPU VPs and 110 shared-memory users, and you intend to add two CPU VPs dynamically as needed, you must include at least five semaphore sets in the SEMMNI parameter: one set for the initial VPs, two sets for the dynamically added CPU VPs, and two sets for the shared-memory connections. You must set the SEMMSL parameter to at least 100. If your system requires a value for the SEMMNS parameter, you must indicate a total of no less than 133 (15 + 2 + 110 + 6).

If your system uses software packages that require semaphores in addition to the ones that your target database server needs, you must include the total number of semaphore sets that are required by both the database server and your other software packages in the SEMMNI parameter. You must set the SEMMSL parameter to the largest number of semaphores per set that any package requires. For systems that require the SEMMNS parameter, you can multiply SEMMNI by the value of SEMMSL to calculate an acceptable value. Or, to arrive at a more precise value for SEMMNS, you can calculate the number of semaphores that are needed for each software package and add those numbers to obtain the total.

Configuring the Number of Open File Descriptors

Some operating systems require you to specify a limit on the number of file descriptors that a process can have open at any one time. You specify this limit with an operating-system configuration parameter, typically NOFILE, NOFILES, NFILE, or NFILES. The number of open file descriptors that each instance of your target database server needs is the number of chunks in your database plus the number of network connections that your database server instance must support.

Operating-System Updates

The target database server might require you to install operating-system updates or *patch releases*. For information about operating-system patches that your target database server installation might require, refer to the machine-notes file.

Disk-Use Issues

When you convert from a pre-6.0 version of the database server, you must allow for increased disk use both during and after the migration process. Some disk resources that must be allocated to the root dbspace during the migration process can be freed for other uses after the process completes. The additional disk space that is required falls into the following categories:

- Space for the conversion processes
- Space in each index entry
- Space for data distributions

The initial requirements during conversion include 1,100 additional pages in the root dbspace over its size under pre-6.0 versions of the database server. The database server uses these added pages to build the **sysmaster** database. You must also provide additional space in the root dbspace for automatic conversion of system catalog indexes.

An additional byte per index entry for each user-table index must be allocated to the dbspace in which each corresponding table resides.

If you intend to use data distributions, you must provide enough temporary space to hold the largest table for which you intend to establish a distribution. For information about data distributions, refer to UPDATE STATISTICS in the *Informix Guide to SQL: Syntax*.

The remainder of this section outlines the additional hard-disk requirements for migration to your target database server. You need to examine carefully the amount of disk space left in each dbspace.

The first time that you start up your target database server, the database server performs the following tasks automatically:

- Conversion of system catalog indexes to your target database server, (occurs at the start of quiescent mode)
- Creation of **sysmaster** and **sysutils** databases (occurs at the start of on-line mode)

You must provide enough space for these actions to take effect. After the database server begins normal operation, you must rebuild the indexes for user tables. You must allocate adequate disk resources for those indexes as well.

Accommodating System Catalog Indexes

The following formula indicates how many additional pages you must provide to accommodate the growth in system catalog indexes for a given database:

```
growth_in_pages = (Total * .10) + (Largest * 1.10)
```

Total is the total number of leaf pages for all system catalog indexes.

Largest is the number of leaf pages in the largest index.

The additional space that is available during the conversion process must include room for the following items:

- A copy of the largest index plus 10 percent
- 10 percent of the current total of system catalog index pages

The database server requires this amount of additional space to accommodate those brief periods in which the old and new versions of an index both reside on disk. The additional 10 percent allows for the case in which the largest index also is the last one to be converted.

You can use the following SQL query within DB-Access or INFORMIX-SQL to determine how many added pages the new system catalog indexes require:

```
UPDATE STATISTICS;
SELECT ((SUM(leaves) * 0.10) + (MAX(leaves) * 1.10)) sci_added
    FROM sysindexes
    WHERE tabid < 100;
```

Important: *You must perform this query from OnLine Dynamic Server 4.1 or 5.0.*

If sufficient space is not already available in the root dbspace, you must allocate additional chunks or move tables to other dbspaces to make room. You can use the **tbstat –d** command to find the number of free pages in the root dbspace. For details, refer to your *Administrator's Guide*, Version 4.1 or 5.0.

Accommodating the sysmaster Database

The **sysmaster** database is created in the root dbspace and cannot be moved or redirected. The **sysmaster** database contains *pseudotables* for monitoring and real tables to store backup information for ON-Archive.To create the **sysmaster** database you need up to 1,100 free pages in the root dbspace.

Locating Temporary Files and Tables

Version 6.0 and later versions of the database server let you use the DBSPACETEMP configuration parameter or the **DBSPACETEMP** environment variable to specify the location of temporary files and tables in either raw or cooked space in UNIX. When you use Windows NT, you can specify the locations of temporary files and tables in buffered and unbuffered space. (Previous versions of the database server create temporary tables in the root dbspace by default. Version 6.0 and later versions of the database server do not use the **DBPATH** environment variable to locate sort files.) If your computer has at least two hard disks, you might consider mirroring your root dbspace and redirecting the temporary table creation elsewhere. This setup prevents hard-disk failures on the root dbspace from affecting your day-to-day business activities.

Accommodating Data Distributions

If you intend to use data distributions, you must provide adequate space for them in the dbspace that contains the system catalog tables for each database in which they are used. (Use the UPDATE STATISTICS statement, described in the *Informix Guide to SQL: Syntax*, to create data distributions.) The following formula gives the maximum amount of space in bytes that might be required for a data distribution on an individual column. Add the results for each column to obtain the total amount of additional space that is needed for the distributions themselves:

```
dist_space = (ceil((4 * ((1/d_res) + 1) *
    (4 + c_len))/765) * 1,116) + 2
```

ceil represents a mathematical function that rounds its argument to the next larger integer. On many systems, this function is called *ceil* or *ceiling*.

d_res is the decimal representation of the resolution that is specified in the UPDATE STATISTICS statement. The default resolution for a HIGH-mode data distribution is 0.5 percent, or 0.005 in this formula.

c_len is the length in bytes for the column. A FLOAT column typically contains 8 bytes; a CHAR 20 column contains 20 bytes.

Tip: *This formula yields the maximum possible size for a distribution that contains a number of overflow entries. A typical distribution with few or no overflow entries uses only 25 percent of the maximum space that this formula projects.*

For MEDIUM-mode distributions, you must also provide sort space in the dbspace equivalent to 3,000 rows of the widest table. For HIGH-mode distributions, you must provide space for a complete copy of the largest table for which you want a HIGH-mode data distribution.

For example, if you intend to use a HIGH-mode data distribution with the default resolution of 0.5 percent on a CHAR 20 column, the following calculation shows the maximum space that is needed for that distribution in bytes:

```
(ceil((4 * ((1/0.005) +1) * (4 + 20))/765) * 1,116) + 26
```

This formula works out to 29,042 bytes, as shown by the following calculations:

```
(ceil((4 * (200 +1) * 24)/765) * 1,116) + 26
(ceil((4 * 201 * 24)/765) * 1,116) + 26
(ceil(19,296/765) * 1,116) + 26
(ceil(25.223) * 1,116) + 26
(26 * 1,116) + 26
29,016 + 26
```

If this column appears in a table that contains 100,000 rows of 28 bytes each, you must provide a minimum of approximately 2.8 megabytes of sort space in the dbspace to build this distribution.

For a MEDIUM distribution, calculate the required amount of sort space as follows:

```
sort_space = 28B * 3,000 = 84,000B
```

Accommodating User-Table Indexes

The following formula indicates the number of additional pages that are required to accommodate the growth in user-table indexes for a given database:

```
index_growth_pages = (total * 0.10)
```

total is the total number of leaf pages for all user-table indexes.

You might need to add chunks to your existing dbspaces, or perhaps add new dbspaces and move tables to that dbspace to provide additional room. You can use the **tbstat –d** command to find out the number of free pages in the current dbspace. For details, refer to your *Administrator's Guide* (Version 4.1 or 5.0).

You can use the following SQL query in DB-Access or INFORMIX-SQL to determine the number of additional pages that user-table indexes need for an entire database:

```
SELECT (SUM(leaves) * 0.1) uti_added
    FROM sysindexes
    WHERE tabid >= 100;
```

Indexes reside in the same dbspace as the tables to which they refer. Tables can be located in different dbspaces than the databases in which they are managed. If all of your tables reside in the current dbspace, the result of this query indicates the number of pages to add to the database. However, if one or more tables reside in separate dbspaces, you must make sure that those dbspaces include enough room for the new indexes.

To find out the dbspace in which each external table resides, use the following SQL query:

```
SELECT tabname, tabid, (partnum / "0x100000") dbspace_num
    FROM systables
    WHERE tabid >= 100
    AND partnum > 0;
```

For each table that resides in a separate dbspace, use the following SQL query to find out how many additional index pages must be added to the estimate for its dbspace. Add the resulting number of pages to the estimate for that dbspace, and deduct it from your estimate for the current dbspace:

```
SELECT (leaves * 0.1) tbl_added
    FROM sysindexes
    WHERE tabid = alt_tabid;
```

alt_tabid is the tabid (table ID number) of the table in a separate dbspace as returned by the previous query.

Accommodating the Conversion of User-Table Indexes

The process of converting user-table indexes by your target database server to use is not automatic and requires preliminary planning. You convert user indexes in the last steps of the migration procedure, after the target database server is installed. For more information, refer to "Use the oncheck Utility to Convert Indexes" on page 6-35.

You can choose from three different methods to convert indexes for a user table. The method that you choose for a given table depends on the size of the table, the degree to which availability of the table is seen as critical, the logging mode of the database, and the time that you can allow for your target database server to stay closed to other users.

You can use the following conversion methods:

- The **oncheck –cI –y** command
- The UPDATE STATISTICS statement
- The DROP INDEX and CREATE INDEX statements

Tip: *The time you need to convert user-table indexes is proportional to the time you need to rebuild indexes in pre-6.0 versions of the database server. Factors that affect the conversion time include the number of indexes per table, the size of each index, the number of available CPUs, and the absence or presence of other user activity on the system.*

The **oncheck –cI –y** command converts indexes while the database is in quiescent mode. This method is preferred for converting large or critical tables. You can also use this method to convert an entire ANSI-compliant database in a single operation. Usually, this method is also the easiest method to use for any table or database. However, the database server remains unavailable to other users as long as the **oncheck** utility is running. For more information about the **oncheck** utility, refer to your *Administrator's Guide*.

The UPDATE STATISTICS statement provides added flexibility. You can use it to convert indexes while other users are on-line. However, if you execute this statement within a transaction while other users are working, you risk bringing the database server off-line with a long-transaction error. Handle small tables in ANSI-compliant databases one at a time if other users are working on the database server. Do not use this method to convert large or multiple tables in ANSI-compliant databases. For more information about the UPDATE STATISTICS statement, refer to the *Informix Guide to SQL: Syntax*.

Dropping and rebuilding indexes is another conversion option that you can use in place of UPDATE STATISTICS. Because the UPDATE STATISTICS statement also allows you to generate data distributions, it is generally preferred over dropping and rebuilding indexes with the DROP INDEX and CREATE INDEX statements. For more information about these statements, refer to the *Informix Guide to SQL: Syntax*.

To prepare for converting a user index

1. Execute the following query in each database:

   ```
   SELECT tabname
       FROM systables
       WHERE tabid >= 100
   ```

2. Determine which method to use for each table in your list.

   The following chart gives recommendations for tables depending on their size, importance, the type of database in which they reside, and the urgency with which the database server must be brought back on-line. Use **oncheck** to convert large, critical tables. The choice for small, but critical, or large, but noncritical tables depends on which scenario produces the smaller effect: having the database server unavailable, or rebuilding indexes while your system is active.

Type of Table	Needed Quickly	Less Urgent
Large, critical	**oncheck -cI -y**	**oncheck -cI -y**
Small, critical	**oncheck -cI -y**	UPDATE STATISTICS
Large, noncritical	**oncheck -cI -y**	UPDATE STATISTICS
Small, noncritical	UPDATE STATISTICS	UPDATE STATISTICS

Tip: *You might be able to use a single method to convert the indexes on all or most tables within a database. Both the* ***oncheck*** *command and UPDATE STATISTICS statement allow you to use a single command to convert the indexes for an entire database. For details, see the "Informix Guide to SQL: Syntax."*

Managing Secure-Auditing Log Files

If you intend to use the secure-auditing features of the target database server, be advised that the database server audit-record log files can grow rapidly to take up a significant amount of space in the file system. Be sure to allow adequate space for the file system in which these files reside. You can configure audit records to minimize the effect of secure auditing on disk use. For details, refer to your *Trusted Facility Manual*.

Database Server Configuration Issues

This section provides an overview of database configuration issues involved in the migration process. The following discussions describe only those configuration issues that affect the migration process. For detailed information about the database server configuration parameters, refer to your *Administrator's Guide*.

Changing Environment Variables

Version 6.0 and later versions of the database server include new environment variables that replace those in OnLine Dynamic Server 4.1 and 5.0. Environment variable names that began with **TB** in earlier versions begin with **ON** in the database server versions 6.0 or later. For instance, the **TBCONFIG** environment variable is replaced by the **ONCONFIG** variable. The **SQLEXEC** environment variable is not used in the database server, Versions 6.0 and later.

When you initialize an instance of your target database server, you must set the **INFORMIXSERVER** environment variable to the dbservername of that instance. Applications must also set the **INFORMIXSERVER** environment variable to gain access to databases that your target database server manages.

Revising Configuration Parameter Values

The database server, Version 6.0 and later, recognize new minimum values for certain configuration parameters that existed prior to Version 6.0.

As of Version 7.1, the BUFFERS parameter indicates the maximum number of buffers for disk I/O. To improve performance, increase the BUFFERS and DBSPACETEMP values, if necessary.

You must increase the value of the LOCKS parameter to at least 2,000. The database server puts a message in the message log if more locks are needed.

Increasing Logical-Log Space

Make sure that at least 2,000 total log pages are allocated and free for logical logs because the building of the **sysmaster** database requires 1,000 log pages. Informix recommends a 1,000 log-page safety factor. Run **tbstat -l** for your current log-usage status.

Planning for Additional Shared-Memory Usage

The database server architecture, introduced with Version 6.0, combines all the memory that **sqlturbo** processes use in previous versions of the database server into a shared-memory section called the *virtual segment* of shared memory. When you migrate from OnLine Dynamic Server 4.1 or 5.0, you must allocate enough shared memory with the SHMVIRTSIZE configuration parameter, in the ONCONFIG file, to accommodate the user threads that separate database server processes serviced in pre-6.0 versions. A reasonable initial estimate for SHMVIRTSIZE is 500 kilobytes for each user thread. Additional space might be required for use with data distributions. Use the SHMVIRTSIZE configuration parameter to specify the amount of shared memory for the database server.

The database server requires three segments of shared memory as opposed to the one segment that was required in pre-6.0 versions of the database server. For information about how to configure shared-memory segments in the operating system, refer to "Estimating the Size and Number of Shared-Memory Segments" on page 6-6.

You use the SHMADD parameter in the ONCONFIG file to set the size of a dynamically added segment. If you do not specify a value for this parameter, the database server attempts to attach additional shared memory in 8-megabyte segments.

The SHMTOTAL parameter in the ONCONFIG file places an absolute maximum on the amount of shared memory that an instance of the database server, Version 7.3, can request. To avoid the risk of exceeding the shared memory provided for the database server, set this parameter to the maximum amount of shared memory for that instance requires. If you set SHMTOTAL to 0 or leave it unassigned, the database server continues to attach additional shared memory as needed until no more virtual memory is available on the system.

Saving Pre-Existing sysmaster Databases

Pre-6.0 versions of the database server include the **makeps.sql** script, which creates a **sysmaster** database. If this database exists on your system, you can run **dropps.sql** to drop it. If you do not drop the old **sysmaster** database, the conversion process renames it **sysmaster_pre60**.

Configuring Secondary Database Servers for Data Replication

When you configure a secondary database server for use in data replication, the version of your target database server on the secondary host computer must match that on the primary host computer. Chunk names and offsets must also match between instances of your target database server on the primary and secondary hosts. For more information about data replication, refer to your *Administrator's Guide*.

Managing Backups

The ON-Archive backup-management system, introduced in OnLine Dynamic Server 6.0, and table fragmentation, introduced in OnLine Dynamic Server 7.1, might change the placement of databases and tables on disk.

The ON-Archive menu uses a special termcap file that is located in the **tctermcap** file in **$INFORMIXDIR/etc**. If your terminal is not listed in this file, you might need to add a new entry to use the menu interface. For more information about backup strategies, table organization, and the **tctermcap** file, refer to refer to your *Administrator's Guide*.

Making Performance Comparisons

Informix recommends that you run and record time- and resource-use statistics for sample queries and other operations to help you compare performance before and after migration to your target database server. You can compare these statistics with equivalent operations you perform after conversion to characterize performance enhancement or degradation. The comparison might help you identify database-configuration parameters that you can adjust to obtain better performance.

UNIX

Migration from OnLine 4.1 or 5.0 to the Database Server 6.0 or Later

The procedure for migrating from a pre-6.0 database server to the database server, Version 7.3, on UNIX includes the following parts. Each process includes a number of individual steps, as the following sections outline:

1. Install the latest maintenance release for the current version.
2. Capture configuration and chunk-layout information in OnLine Dynamic Server, Version 4.1 or 5.0.
3. Close all transactions and make a final (level-0) backup under OnLine Dynamic Server, Version 4.1 or 5.0.
4. Bring OnLine Dynamic Server, Version 4.1 or 5.0 off-line.
5. Install and configure your target database server.
6. Reconfigure the operating system.
7. Bring your target database server on-line.
8. Convert user-table indexes.
9. Verify the integrity of the database.
10. Make an initial (level-0) backup under your target database server.

Install the Latest Maintenance Release for the Current Version

Informix recommends that you install the latest maintenance release for your current database server version before you migrate to a new version, especially if you are using Version 5.0. In this scenario, you plan to migrate from OnLine Dynamic Server 5.03 to Informix Dynamic Server, Version 7.3, on UNIX. First, install the latest maintenance release for your database server, then migrate to Version 7.3. Many minor changes to the 5.x versions are also in the 7.x versions.

Capture Configuration and Chunk-Layout Information

Before you can safely configure your target database server, you must capture the configuration and chunk-layout information under OnLine Dynamic Server 4.1 or 5.0. You use this information when you configure your target database server.

To copy database server configuration files

1. Log in as user **informix**.
2. Ask all OnLine Dynamic Server 4.1 or 5.0 users to exit their applications.
3. Rename or make a copy of the configuration file(s) if the current name appears in the following list:
   - **$INFORMIXDIR/etc/onconfig**
   - **$INFORMIXDIR/etc/onconfig.std**
   - **$INFORMIXDIR/etc/sqlhosts**
   - **$INFORMIXDIR/etc/tbconfig**
   - **$INFORMIXDIR/etc/tbconfig.std**
   - **$INFORMIXDIR/etc/tctermcap**
   - **$INFORMIXDIR/etc/termcap**

   You must save these files because the installation procedure for your target database server overwrites them during the installation. Keep the copies available to use later.

To copy chunk-layout and space-use information

Execute the following command to save a listing of your chunk layout and space use in case you need to refer to it:

```
tbstat -d > $INFORMIXDIR/chunk.layout
```

To add chunks or to move tables to accommodate your target database server disk-use requirements

If you need to add chunks or move tables out of the root dbspace to make room for the **sysmaster** database and system catalog indexes, you can do so at this time. You can also add chunks or move tables in this and other dbspaces to accommodate user-table indexes.

For more information about disk-space requirements in your target database server, refer to "Disk-Use Issues" on page 6-13. For information about adding chunks or moving tables, refer to the 4.1 or 5.0 version of your *Administrator's Guide*.

Close All Transactions and Make a Final Level-0 Backup

Communicate to client users how long you expect the database server to be off-line for migration. This procedure prepares the transaction log for migration to your target database server, and creates a final backup of the database under OnLine Dynamic Server 4.1 or 5.0.

To perform an immediate shutdown

Execute the following command to perform an immediate shutdown:

```
tbmode -k
```

Answer `yes` to all the prompts. This step terminates all database server processes that might still be running.

To shut down the system gracefully

1. Execute the **tbmode –sy** command.
2. Wait for all users to exit.
3. Execute the **tbmode –k** command; then answer `yes` to all the prompts.

Make sure the system has shut down completely before you proceed to the next step.

To initiate a fast recovery

Execute the following command to enter quiescent mode and initiate a fast recovery:

```
tbinit -s
```

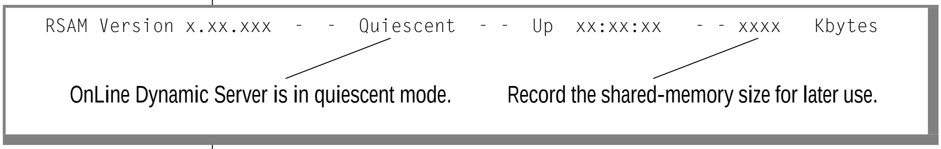
The **tbinit –s** option rolls forward all committed transactions, rolls back all incomplete transactions since the last checkpoint, and leaves a new checkpoint record in the log with no open transactions pending. (Refer to the 4.1 or 5.0 version of your *Administrator's Guide*.)

To verify the operating mode

Execute the following command to verify that OnLine Dynamic Server 4.1 or 5.0 is in quiescent mode.

```
tbstat -
```

The first line of the **tbstat** output contains the status of OnLine Dynamic Server 4.1 or 5.0. Figure 6-12 shows that OnLine Dynamic Server 4.1 or 5.0 is in quiescent mode.



```
RSAM Version x.xx.xxx  -  -  Quiescent  - -  Up  xx:xx:xx   - - xxxx   Kbytes
```

OnLine Dynamic Server is in quiescent mode.

Record the shared-memory size for later use.

Figure 6-2
Example of tbstat Status Line

To verify the integrity of the data

Before you make a level-0 backup, you might want to verify the integrity of the data.

Use the **tbcheck** utility to verify the integrity of data before you start the backup. You can verify the integrity of the reserve pages, system catalog tables, data, and indexes. (Refer to the 4.1 or 5.0 version of your *Administrator's Guide*.) Execute the following commands.

To check	Use the command
Reserve pages	`tbcheck -cr`
System catalog tables	`tbcheck -cc` *database_name*
Data	`tbcheck -cD` *database_name*
Indexes	`tbcheck -cI` *database_name*

To make a final backup with OnLine 4.1 or 5.0

Use **tbtape** or **tbmonitor** to make a level-0 backup of OnLine Dynamic Server 4.1 or 5.0 data. The tape parameters must specify a valid tape device. Be sure to retain and properly label the tape volume that contains the backup. (For more information on making a level-0 tape backup, refer to the 4.1 or 5.0 version of your *Administrator's Guide*.)

Important: *Backups that you made under OnLine Dynamic Server 4.1 and 5.0 are not compatible with your target database server tape format. Use only the older version of the source database server to restore the backup tape that you made in this step. You cannot use this backup tape with your target database server.*

Bring OnLine 4.1 or 5.0 Off-Line

This procedure shuts down the instance of OnLine Dynamic Server 4.1 or 5.0.

To shut down the instance of OnLine Dynamic Server

Execute the following command:

```
tbmode -k
```

Answer `yes` to all prompts.

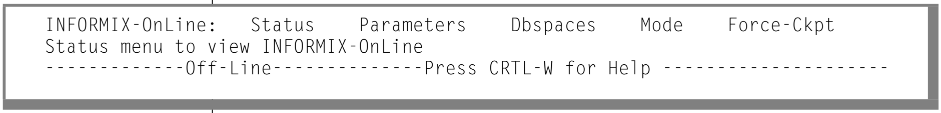
To verify that OnLine Dynamic Server 4.1 or 5.0 is off-line

Execute the following command to verify that OnLine Dynamic Server 4.1 or 5.0 is in off-line mode:

```
tbmonitor
```

Tip: *Use **tbmonitor** instead of **tbstat** to verify the operating mode. The **tbstat** utility is not designed to return the operating-mode status when the database server is off-line.*

The third line of the DB-Monitor main menu contains the status of OnLine Dynamic Server 4.1 or 5.0. The screen in Figure 6-13 indicates that the database server is off-line. (Refer to the 4.1 or 5.0 version of your *Administrator's Guide*.)



```
INFORMIX-OnLine:   Status    Parameters    Dbspaces    Mode    Force-Ckpt
Status menu to view INFORMIX-OnLine
-------------Off-Line--------------Press CRTL-W for Help ---------------------
```

Figure 6-3
DB-Monitor Main Menu

The database server must be off-line because the older and the newer versions share common files. You cannot install the database server if any of the common files are active. Bring the database server off-line to ensure that all common files are inactive.

Important: *Repeat the previous three sections (pages 6-25 through 6-28) for each database server instance that you want to upgrade.*

Install and Configure the Target Database Server

You must be user **root** to install the target database server. Set the **$INFORMIXDIR** environment variable to the directory where you plan to install the database server.

Follow the directions in your *Installation Guide* and your *Administrator's Guide* to install the target database server. The installation script installs the database server into the **$INFORMIXDIR** directory specified for user **root**. The installation script does not bring the database server on-line.

Warning: *If you install the target database server in the same directory where the older version of the database server currently resides, the newer version overwrites the older files. If you wish to preserve your product files of earlier versions, you must install your target database server in a different directory.*

Before you overwrite the older version, you must take the following precautions:

- If you do not have the original media for the older version, back up the **$INFORMIXDIR** directory before you install your target database server.
- Copy the configuration file(s) in **$INFORMIXDIR/etc** to another location on the file system.

When the installation is complete, exit as user **root** and log in as user **informix**.

Important: *The target database server includes networking capabilities that are not present in pre-6.0 versions. These capabilities use networking information from configuration files such as the ONCONFIG and **sqlhosts** files to establish communications with application processes. You must ensure that these files are present and contain the needed information. To verify these files, establish a connection from DB-Access to test the database server instance before you proceed with your installation. If your configuration calls for it, you can also test a remote client-server system.*

To set up the configuration file

Copy the standard Version 7.3 configuration file to the name of your working configuration file, and then edit it. For example, if your configuration file name is **onconfig.1**, you might use the following commands:

```
cd $INFORMIXDIR/etc
cp onconfig.std onconfig.1
vi onconfig.1
```

Refer to the backed-up copy of the OnLine Dynamic Server 4.1 or 5.0 configuration file that you saved in "Capture Configuration and Chunk-Layout Information" on page 6-25. Copy the values from matching parameters in the backed-up file to your working configuration file (for example, **onconfig.1**). Modify those values to conform with revised minimums for Version 7.3 as described in "Revising Configuration Parameter Values" on page 6-21.

To set environment variables for the target database server

At this point, you must set up the environment variables that both the client applications and your target database server need to communicate. Make sure that the **INFORMIXDIR** and **PATH** environment variables were set during the installation. (For more information, refer to your *Administrator's Guide*.)

Set the **ONCONFIG** environment variable to the name of your target database server configuration file, as shown in the following examples:

C shell:

```
setenv ONCONFIG onconfig.1
```

Bourne shell:

```
NCONFIG=onconfig.1;
export ONCONFIG
```

Set the **INFORMIXSERVER** environment variable for all users that need to access the target database server. Set this variable to the name that is listed in the **sqlhosts** file, and in the DBSERVERNAME or DBSERVERALIASES parameter of your target database server configuration file, as follows:

C shell:

```
setenv INFORMIXSERVER dbserver1
```

Bourne shell:

```
INFORMIXSERVER=dbserver1;
export INFORMIXSERVER
```

If you use a dbspace or file to store temporary tables, set the DBSPACETEMP configuration parameter or **DBSPACETEMP** environment variable to the name of the dbspace or full pathname of that file.

To modify the sqlhosts file

The target database server requires an **sqlhosts** file. An **sqlhosts.demo** file is included in **$INFORMIXDIR/etc** as an example of the setup that your target database server requires. If you do not have an **sqlhosts** file already, the installation program renames the **sqlhosts.demo** file to **sqlhosts**. The **sqlhosts** file should include an entry with the following information for each instance of your target database server:

```
dbservername   nettype   hostname   service_name   options
```

You must modify the entries in this file to support your configuration. For more information on how to modify the **sqlhosts** file, refer to your *Administrator's Guide.*

Important: *The client application looks for the **sqlhosts** file in the **$INFORMIXDIR/etc** directory. However, you can use the **INFORMIXSQLHOSTS** environment variable to change the location or name of the **sqlhosts** file.*

Reconfigure the UNIX Operating System

For this step you need to reconfigure your operating system based on the estimates that you determined in "Planning for Migration" on page 6-6. You need to provide additional shared memory, additional semaphores, and possibly additional hardware resources such as disk drives. To reconfigure the operating system, follow the directions in the machine-notes file included in your target database server distribution and the kernel-configuration instructions for your operating system.

Bring the Target Database Server On-Line

This step brings the database server to quiescent mode. The success of this step depends on adequate operating-system and disk resources, as discussed in "Planning for Migration" on page 6-6. If the system is not brought up in quiescent mode, you get the following error when you attempt to initialize your target database server and the database server goes off-line:

```
Open transaction detected when changing log versions.
```

To bring the target database server from off-line to quiescent mode

To bring your target database server from off-line to quiescent mode, execute the following command:

```
oninit -s
```

Execute the **onstat -m** command to check the message log for the status of the mode change and to create the **sysmaster** database.

The system automatically creates the **sysmaster** database when the target database server is brought on-line. If the system fails to create this database, the root dbspace might not include the additional 550 pages that are needed for conversion. Return to "Install and Configure the Target Database Server" on page 6-29, provide enough space in the root dbspace, and repeat the steps in this section. If the **sysmaster** database does not already exist, the database server keeps trying to create it each time that you start the database server.

Warning: *The logical logs continue to fill with the transactions that result from the creation of the* ***sysmaster*** *database. If you run out of log space before the creation of the* ***sysmaster*** *database is complete, the database server halts with a long-transaction error. Thus, you must back up the logical logs.*

Execute the **ontape -a** or **ontape -c** command. Once the logical logs are backed up, processing resumes.

When the system reaches quiescent mode successfully, you can note whether you need to adjust the shared memory and semaphore values for your operating system, which you configured in "Reconfigure the UNIX Operating System" on page 6-31. Check your database server message log for status messages that pertain to the change to quiescent mode.

To bring the target database server from quiescent to on-line mode

To change your the database server mode from quiescent to on-line, execute the following command:

```
onmode -m
```

At this point, the system attempts to rebuild system catalog indexes. If you try to access them at this time, you might find some of them locked. If these catalogs cannot be rebuilt, you might need to allocate more space in your root dbspace. Along with the 1,100 extra pages that you need to create the **sysmaster** database, you must add the number of additional pages that the formula gives in "Accommodating System Catalog Indexes" on page 6-14. Return to this section, and allocate sufficient space.

Convert User-Table Indexes

If you have not already done so, use the formula provided in "Accommodating User-Table Indexes" on page 6-17 to allocate the additional disk space that you need for user-table indexes in your target database server. You can now begin converting user-table indexes.

Converting user-table indexes requires planning. You can use several methods to convert indexes for various tables. Depending on the size and demand for a table, you might prefer one method over another. If you have not already done so, generate a list of tables in each of your databases and identify the appropriate conversion method for each index, as described in "Accommodating the Conversion of User-Table Indexes" on page 6-18.

Before you make the database server available to regular users, execute the **oncheck -cI -y** command to convert indexes for large or critical tables, as described in "Use the oncheck Utility to Convert Indexes" on page 6-35. You can defer smaller and less-critical tables until after the database server is brought on-line. You can then convert these tables through SQL with the UPDATE STATISTICS statement, or the DROP INDEX and CREATE INDEX statements.

You might be able to expedite the process of converting indexes by taking advantage of enhanced features that are provided with your target database server such as support for multiple CPUs, parallel index builds, and so on. For more information, refer to your *Administrator's Guide*.

The new indexes will be larger than the older indexes no matter which method you choose. Make sure that you allocate enough room in each dbspace for the new indexes. Otherwise, you might need to move tables to other dbspaces before you convert indexes, as described in the following section, "Move Tables to Another dbspace."

Move Tables to Another dbspace

If you neglected to make the proper adjustments to your dbspaces as described under "Accommodating User-Table Indexes" on page 6-17, you might need to move one or more tables to another dbspace to accommodate the larger index in your target database server. If you know that you have enough room for your new indexes, you can skip ahead to "Use the oncheck Utility to Convert Indexes."

You can use either of the following techniques to move tables during the conversion process:

- The INSERT statement
- The **onunload** and **onload** utilities

To use INSERT to move tables to a new dbspace

1. Create a table in the new dbspace with a temporary name.
2. Use INSERT INTO *newtab* SELECT * FROM *oldtab*.
3. Drop the old table.
4. Create all the indexes in the new table.
5. Rename the new table to the original name.

To use onunload and onload to move tables to a new dbspace

1. Create a new table in another dbspace.
2. Drop all the indexes from the old table.
3. Use **onunload** to copy the old table to a file.
4. Drop the table, then update the database to create a new version of the table in the new dbspace.
5. Use **onload** to load the data back into the table.
6. Create all the indexes in the new table.

Use the oncheck Utility to Convert Indexes

The **oncheck** utility can be helpful when you convert ANSI-compliant databases, and in situations where critical tables must be made accessible as soon as possible.

You can run **oncheck –cI –y** to convert indexes for a single table or an entire database when your target database server is in quiescent mode. After you make sure that no users are present, use **onmode –s** to bring the system into quiescent mode.

To convert indexes for a single table

```
oncheck -cI -y database_name:table_name
```

database_name is the name of the database.

table_name is the name of the table.

To convert indexes for the entire database

```
oncheck -cI -y database_name
```

The **oncheck** command displays messages of the following form:

```
Index index-name is bad. OK to repair it?
```

This message indicates that the existing (4.1 or 5.0) index is not in the correct format for your target database server. The **–y** option automatically answers `yes` to this prompt, allowing your target database server to convert each index in turn automatically.

Use UPDATE STATISTICS to Convert Indexes

Tip: *The UPDATE STATISTICS statement includes new options that were introduced in OnLine Dynamic Server 6.0. For details, refer to the "Informix Guide to SQL: Syntax."*

You can execute an UPDATE STATISTICS statement to convert the indexes for a single table or, in some circumstances, for an entire database. The target database server UPDATE STATISTICS statement automatically converts Version 4.1 and 5.0 indexes, provided that the dbspace has enough space. For more information, see "Move Tables to Another dbspace" on page 6-34.

This method is especially useful for nonlogging databases. Or you can use UPDATE STATISTICS in a database without ANSI logging by issuing the statement *outside of a transaction*. When you take the proper precautions, you can even use this method within a transaction or an ANSI-compliant database.

Warning: *If you execute an UPDATE STATISTICS statement within a transaction, you must follow up with a COMMIT WORK or ROLLBACK WORK statement to close the transaction. Otherwise, the database server, Version 7.3, instance eventually halts with a long-transaction error.*

When you convert indexes within a transaction or an ANSI-compliant database, Informix recommends that you use one of the following approaches to limit the risk of encountering a long transaction:

- Limit the scope of each UPDATE STATISTICS statement to a single table. This approach reduces your risk of encountering a long transaction. Execute separate UPDATE STATISTICS statements for each table, followed by separate COMMIT WORK statements.
- Update statistics for an entire database only when no other users have access to the current instance of your target database server.
- Use **oncheck** to convert indexes while the database server is in quiescent mode. For more information, see "Use the oncheck Utility to Convert Indexes" on page 6-35.

Use DROP INDEX and CREATE INDEX to Convert Indexes

If neither of the previous two methods seem suitable, you can drop and rebuild the indexes for your tables individually. When you drop and rebuild indexes, you can override the default fill factor that is specified in the FILLFACTOR parameter of the ONCONFIG file. For details, refer to the *Informix Guide to SQL: Syntax*.

Verify the Integrity of the Data

Informix recommends that you verify the integrity of your data before you run the reversion utility. Figure 6-14 lists the **oncheck** commands for verifying data integrity.

Figure 6-4
Commands for Verifying Data Integrity

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *database_name*
Check data	`oncheck -cD` *database_name*
Check indexes	`oncheck -cI` *database_name*

Tip: *You might see the following warning. It means that you have not defined any synonyms.*

```
WARNING: No syssyntable records found.
```

Make an Initial Backup Under the Target Database Server

Use your target database server backup tool (ON-Bar, ON-Archive or **ontape)** to make a level-0 backup. Do not overwrite the tapes you used earlier when you made your final backup of your database server. For more information, refer to the *Archive and Backup Guide for Informix Dynamic Server* or the *Backup and Restore Guide for Informix Dynamic Server*.

Important: *Do not restore the backed up logical logs from your source database server to the newer version of your database server.*

Add Rowids to Fragmented Tables

In the database server, Version 7.x and later, fragmented tables do not have a rowid field by default. If you migrate fragmented data tables to Version 7.3, you must add a rowid column to each table before you use applications that use rowids to access data.

Migration Complete

When you finish the level-0 backup, the migration process is complete and users can use your database server to access data safely.

The first time you bring up the target database sever, the **sysmaster** database is built in 15 to 30 minutes. Check the message log to ensure that the **sysmaster** database build has completed before you allow users to access the database server.

Once you successfully migrate to your target database server, you might want to seek ways to obtain maximum performance. If you created sample queries for comparison, you can use them to characterize the performance differences between your earlier version and your target database server. The results of these comparisons might suggest adjustments to configuration parameters, or to the layout of databases, tables, and chunks. For details on topics related to performance, refer to your *Performance Guide*.

GLS

Global Language Support

When you complete the migration from OnLine Dynamic Server 4.1 or 5.0 to Informix Dynamic Server, Version 7.3, your databases use the default locale, U.S. English. When the database server, Version 7.3 is running successfully, you can take the following steps to migrate a database to a non-default locale.

To convert a database to a non-default locale

1. Use the UNLOAD statement or the **dbexport** utility to unload data from the database to be converted.
2. Set the **CLIENT_LOCALE** and **DB_LOCALE** environment variables to support the new GLS locale.

3. Create a database with the new locale by issuing the CREATE DATABASE statement from an application that has the proper client locale variables set in its environment.
4. Modify the schema representation to replace CHAR and VARCHAR columns with NCHAR and NVARCHAR columns, respectively.
5. Use the LOAD statement or the **dbimport** utility to load the data into the new database.

For more information about the GLS feature, refer to “Changing Locales” and to the *Informix Guide to GLS Functionality*.

Reverting to OnLine 5.0

This section describes the process to revert from Informix Dynamic Server, Version 7.3, to OnLine Dynamic Server, Version 5.0. These steps also apply if you are reverting from Versions 7.2x, 7.10.UD1, 7.10.UCx, or 6.0. Before you can revert from Informix Dynamic Server, Version 7.3, to OnLine Dynamic Server 5.0, you must modify the configuration limits and remove constructs that OnLine Dynamic Server 5.0 does not support.

The procedure for reverting to OnLine Dynamic Server includes the following steps, which are described in the next sections:

1. Save copies of your database server configuration files.
2. Remove database server users.
3. Verify the integrity of the data.
4. Back up database server data.
5. Remove features added by later versions.
6. Execute the reversion utility (**onmode -b**).
7. Prepare the TBCONFIG configuration file.
8. Reset environment variables.
9. Bring up OnLine Dynamic Server 5.0.
10. Verify the integrity of the data.
11. Back up OnLine Dynamic Server 5.0.
12. Bring OnLine Dynamic Server 5.0 into on-line mode.

Save Configuration Files

Before you start the reversion process, save copies of your configuration files.

Remove All Users from the Database Server

Remove all users from the database server before you begin the reversion process. Warn the users that you plan to shut down the database server, and then execute the following command:

```
onmode -s
```

The **-s** flag on **onmode** restricts new access to the database server but allows current processing to finish. When all processing is finished, the database server goes to quiescent mode, and you can continue the reversion process.

Verify the Integrity of the Data

Before you allow users to access the databases, use the **oncheck** utility to verify that no data was corrupted in the migration process. You can verify the integrity of the reserve pages, system catalog tables, data, and indexes. For more information, see "Verify the Integrity of the Data" on page 6-37.

Back Up the Database Server Files

Informix recommends that you use **ontape** or ON-Archive to make a level-0 backup. For details on how to make backups, refer to the *Archive and Backup Guide for Informix Dynamic Server*.

Remove Features Introduced by Later Versions of the Database Server

Before you can revert to an earlier version of the database server, you must remove features that were introduced by later versions of the database server:

- Remove NLS and GLS databases.
- Disable data replication.
- Change all fragmented tables back into unfragmented tables. See your *Administrator's Guide* for instructions on how to defragment tables.
- Disable role separation.
- Remove unsupported SQL statements.
- Adjust configuration parameters.

Figure 6-5 and Figure 6-6 on page 6-42 show configuration parameter limits for Version 5.0 and Version 4.1.

Figure 6-5
Configuration Parameter Limits for Version 5.0

Resource	Parameter	5.0 Limit
Logical logs	LOGFILES, LOGSMAX	<= 1 page of entries
Dbspaces	DBSPACES	<= 40
Chunks	CHUNKS	<= 1 page of entries
Buffers	BUFFERS	<= 512 kilobytes

Figure 6-6
Configuration Parameter Limits for Version 4.1

Resource	Parameter	4.1 Limit
Logical logs	LOGFILES, LOGSMAX	<= 1 page of entries
Dbspaces	DBSPACES	<= 40
Chunks	CHUNKS	<= 1 page of entries
Buffers	BUFFERS	<= 32,000

The chunk limit depends on the length of the pathname that you chose for the chunk. The limit could range from 13 to 58.

Tip: *Logical-log files of differing sizes that were created after 6.0 initialization do not impair the reversion.*

Execute the Reversion Utility

To restore the database server files to a form that is compatible with OnLine Dynamic Server 5.0, execute the following command:

```
onmode -b 5.0
```

The reversion utility includes an implicit **onmode –yuk**. This command forcibly removes all users. After the reversion is complete, the system is in off-line mode.

Tip: *The **onmode -b** command also rebuilds the user-table indexes automatically.*

Prepare the TBCONFIG Configuration File

Modify the configuration file to eliminate parameters that OnLine Dynamic Server 5.0 does not recognize. You might find it easiest to compare your saved configuration file with the **tbconfig.std** file and make adjustments accordingly.

Be sure to include the USERS configuration parameter, which was replaced by USERTHREADS in Version 6.0 and later.

Important: *Use the same values for ROOTPATH, ROOTSIZE, and ROOTOFFSET for both versions of your database server.*

Reset Environment Variables

Reset the environment variables to values that are appropriate for the selected version of your database sever. Remember that in Version 5.0 and 4.1, you specify the configuration file with the **TBCONFIG** environment variable instead of **ONCONFIG**.

The **PDQPRIORITY** environment variable and the SQL SETPDQPRIORITY statement were introduced after Version 6.0. If you set the **PDQPRIORITY** environment variable, it does not cause problems but it might cause confusion. You must remove the SETPDQPRIORITY statement from your applications.

Bring Up the Desired Version of the Database Server

Bring up OnLine Dynamic Server 4.1 or 5.0 from off-line to quiescent mode. Execute the following command:

```
    tbinit -s
```

Verify the Integrity of the Data

Before you allow users to access the databases, check the integrity of the data. Figure 6-17 lists the **tbcheck** commands for verifying data integrity.

Figure 6-7
Commands for Verifying the Data Integrity

Action	oncheck Command
Check reserve pages	`tbcheck -cr`
Check system catalog tables	`tbcheck -cc` *database_name*
Check data	`tbcheck -cD` *database_name*
Check indexes	`tbcheck -cI` *database_name*

When you run **tbcheck**, you might see the following message:

```
OLD pn_bytes != NEW pn_nbytes
```

This message does not require any action on your part. It indicates that a later version of the database server accessed your database.

Back Up OnLine 5.0 Server Data

After you complete the reversion, Informix recommends that you make a level-0 backup. Use the **tbtape** utility to prepare backups. For details on making backups, refer to the appropriate *Administrator's Guide*.

Return OnLine 5.0 to On-Line Mode

To change your OnLine Dynamic Server 5.0 mode from quiescent to on-line, execute the following command:

```
tbmode -m
```

The reversion is now complete, and users can access the converted data.

Chapter

7

Migrating Workgroup Edition and OnLine Workgroup Server

This chapter describes the steps for migrating between Version 7.3 of Informix Dynamic Server, Workgroup and Developer Editions and earlier versions of OnLine Workgroup Server. The term *Version 7.2x* refers to versions 7.2, 7.21, 7.22, 7.23, or 7.24.

This chapter covers the following topics:

- Using different configurations of the database server and the administration tools
- Preparing to migrate
- Upgrading the database server and the administration tools
- Reverting the database server to an earlier version

Tip: *For information on how to move the database server between* UNIX *and Windows* NT *environments, refer to Chapter 9, "Moving Between Database Servers."*

Using Different Server and Client Configurations

This section discusses the effect of different database server and client configurations. Informix supports the following configurations:

WIN NT

- Informix Dynamic Server, Workgroup and Developer Editions, Version 7.3, and Informix Enterprise Command Center, Version 3.0
- OnLine Workgroup Server, Version 7.22 through Version 7.24, and INFORMIX-Command Center, Version 3.0
- OnLine Workgroup Server, Version 7.22, and INFORMIX-Command Center, Version 2.0 ♦
- OnLine Workgroup Server, Version 7.12, and INFORMIX-Command Center, Version 2.0 or Version 1.0

UNIX

- OnLine Workgroup Server, Version 7.2, and INFORMIX-Command Center, Version 1.0
- OnLine Workgroup Server, Version 7.12, and INFORMIX-Command Center, Version 2.0 or Version 1.0 ♦

Workgroup Edition 7.3 and IECC, Version 3.0

Scenario: Upgrade the database server and administration tools. This configuration supports all of the new database server and administration tool functions. If you select **Informix Neighborhood** in the Informix Enterprise Command Center, the details view displays the version number and type of the database server.

OnLine Workgroup Server 7.22 through 7.24 and IECC 3.0

Scenario: Upgrade the database server and administration tools. This configuration supports all of the new database server and administration tool functions. If you select **Informix Neighborhood** in the Informix Enterprise Command Center, the details view displays the version number and type of the database server.

OnLine Workgroup Server 7.12 and IECC 2.0

Scenario: Upgrade the administration tools but not the database server. With the 7.12 database server, you can use all of the Version 2.0 administration tools except **Backup** and **Restore**.

OnLine Workgroup Server 7.22 and IECC 2.0

Scenario: Upgrade the database server but not the administration tools. This scenario could occur when your site has several client computers to upgrade over a period of time. Users would be able to use all of the Version 2.0 administration tools but not GLS on the clients.

OnLine Workgroup Server 7.12 and IECC 1.0

Scenario: You decide not to upgrade the database server or the administration tools.

Preparing to Migrate Between Versions

Use these instructions to migrate to Informix Dynamic Server, Workgroup and Developer Editions, Version 7.3, on Windows NT or to OnLine Workgroup Server, Version 7.2 on UNIX. When you migrate database servers, Informix suggests that you follow these guidelines:

- Review the release notes for all versions of your database server for information about new features, installation, and fixes to problems. Modify applications as needed. Check the documentation notes for information about features that the manuals do not discuss.

  The release notes are in one of the following directories:

  UNIX

  ❑ **$INFORMIXDIR/release/en_us/0333**♦

  WIN NT

  ❑ **%INFORMIXDIR%\release\en_us\0333 (**as of Version 7.2)

  Release Notes appear in the Informix folder. To display this folder, choose **Start→Programs→Informix** from the Task Bar. ♦

UNIX

- Check the machine notes for special actions you need to perform to configure and use your database server on your operating system. ♦

- On UNIX, retain both versions of the Informix product software on disk (if you have enough disk resources). On Windows NT, you cannot retain two versions of the Informix product on disk, prior to database server Version 7.3
- Check the documentation notes for information about features that the manuals do not discuss.
- Retain the installation tapes from both versions of the Informix product software.
- Make a level-0 backup of the database server before and after migration.

Before you upgrade, migrate, or revert your database server, complete the following steps, which are described in the next sections:

1. Check available space and system requirements.
2. Save a copy of the current configuration files.
3. Close all transactions and place the database server in quiescent (administration) mode.
4. Verify the integrity of the data.
5. Back up your database server data.

WIN NT/95

Check Available Space and System Requirements

Before you install your database server and the administration tools, verify that your system meets the minimum space and hardware requirements. Dynamic Server, Workgroup and Developer Editions runs on Windows NT 3.51 and Windows NT 4.0 on an NTFS drive. Informix Enterprise Command Center and Relational Object Manager run on Windows NT 3.51, Windows NT 4.0 and on either a FAT or NTFS drive.

For information on the system requirements, refer to your database server *Installation Guide*, the *Informix Enterprise Command Center Installation Guide*, and the **read_ows.txt** file in Answers OnLine, version 1.7 and earlier.

Save a Copy of the Current Configuration Files

Save a copy of the current ONCONFIG files, located in the **etc** subdirectory of your installation directory, and the **sqlhosts** or registry information:

WIN NT

- **%ONCONFIG%** ♦

UNIX

- **$ONCONFIG**, **osahosts**, **sqlhosts**, **tctermcap**, **termcap** ♦

Save the following files, located in various subdirectories of your installation directory (this step is optional):

- **adtcfg**, located in the **aaodir** subdirectory
- **adtmasks.***, located in the **dbssodir** subdirectory

Place the Database Server in Quiescent Mode

Communicate to client users how long you expect the database server to be off-line for the migration. Terminate all database server processes, and place your database server in quiescent mode (also called *administration mode*).

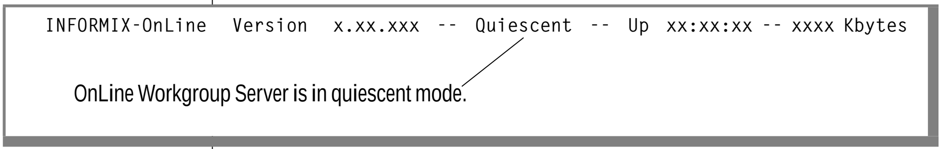
To shut down the database server gracefully

1. Warn all users that you plan to shut down the database server and wait for them to exit.
2. Become user **informix** on UNIX platforms, or on Windows NT, you must be a member of the **Administration** group.
3. Execute the following command to take the database server to quiescent mode:

   ```
   onmode -sy
   ```
4. Wait until your database server is in quiescent mode.

   To verify the mode of your database server, execute the **onstat -** command.The first line of the **onstat** output contains the status of your database server.

   Figure 7-1 shows that the database server is in quiescent mode.



```
INFORMIX-OnLine  Version  x.xx.xxx  --  Quiescent  --  Up  xx:xx:xx -- xxxx Kbytes
```

OnLine Workgroup Server is in quiescent mode.

Figure 7-1
Example of onstat Status Line

5. Execute the following command to force a new logical log:

   ```
   onmode -l
   ```

6. Execute the following command to force a checkpoint:

```
onmode -c
```

7. Execute the following command to shut down the database server:

```
onmode -yuk
```

Tip: *Monitor your log activity to verify that all commands were executed properly and to check for inconsistencies prior to migration.*

To use IECC to place the system in administration mode

1. Select **Administration** from the **Server Mode** list box on the **General** page of the Informix Enterprise Command Center to put the database server in quiescent (administration) mode.
2. Warn all users that you plan to shut down the database server and wait for them to exit. If users are logged on, the Administration wizard lets you notify users when their sessions are to be disconnected.

Warning: *User data can be lost or damaged if you interrupt data transactions. If data transactions are interrupted, shut down and restart the database server in administration mode to initiate a fast recovery. A fast recovery rolls forward all committed transactions and rolls back all incomplete transactions since the last checkpoint and then leaves a new checkpoint record in the log with no open transactions pending.*

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of data before you make a level-0 (complete) backup. If you find any problems with the data, fix them before you make the backup. You can verify the integrity of the reserve pages, extents, system catalog tables, data, and indexes. First, you need to obtain a list of the databases on your database server.

WIN NT

To use oncheck on Windows NT

Open a **Command Line Utilities** window from the **Informix Administration Tools** program group. ♦

Figure 7-2 lists the commands for verifying data integrity.

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *`database_name`*
Check data	`oncheck -cD` *`database_name`*
Check indexes	`oncheck -cI` *`database_name`*

Figure 7-2
Commands for Verifying the Data Integrity

For information on **oncheck**, refer to your *Administrator's Guide.*

Back Up Your Database Server

Make a complete backup of your source database server.

WIN NT

On Windows NT, double-click the **Backup and Restore** tool in the Informix Enterprise Command Center program group. You can also use the command line to back up with **ontape** and ON-Bar on Windows NT. ♦

UNIX

To launch Backup and Restore, enter **$INFORMIXDIR/bin/bar** at the UNIX prompt. You can also use **ontape**, ON-Bar, or ON-Archive to back up on UNIX. ♦

Warning: *The 7.12 and 7.22 and later versions of Backup and Restore are not compatible. The 7.12 version uses **ontape**, but the 7.22 and later versions use **ON**-Bar. Backups that you make under older versions of your database server are not compatible with the newer version. Do not try to restore these backups to the newer version.*

For more information about how to use **ontape**, ON-Archive, or ON-Bar to back up your database server, refer to your *Backup and Restore Guide.*

Preparation Complete

Now you are ready to upgrade, migrate, or revert your database server. For information on how to upgrade, see "Upgrading the Database Server to Version 7.3." For information on how to revert, see "Reverting the Database Server and IECC" on page 7-21.

Upgrading the Database Server to Version 7.3

When you upgrade to Informix Dynamic Server, Workgroup and Developer Editions, Version 7.3, you can install and test the database server with the same database server name, configuration files, environment variables, and shared-server computer that you used for the earlier version. After you install the target database server and verify that it works, you might want to modify the configuration files and environment variables to take advantage of new features.

Complete the following migration steps:

1. Bring the source database server off-line.
2. Install the new version of the target database server.
3. Customize the database server environment. (This step is optional.)
4. WIN NT Configure the database server for Enterprise or Workgroup Replication. (This step is optional.) ♦
5. Bring the target database server on-line.
6. Verify the integrity of the data.
7. Make an initial, complete backup of the database server.
8. Run UPDATE STATISTICS.
9. Verify the access path of your SQL statements.

Save an Output File of SQL Statements for Access Paths

Save a file of output from any SET EXPLAIN statements, so that later you can verify that access paths of your SQL statements do not change when you migrate to your target database server. SET EXPLAIN writes the access path that the optimizer chooses for each query to the SET EXPLAIN output file. The optimizer chooses the fastest path of execution for table joins.

WIN NT

For Windows NT, the SET EXPLAIN output filename is **%INFORMIXDIR%\sqlexpln\*username*.out**. ♦

Bring the Source Database Server Off-Line

Shut down your database server to ensure that all common files are inactive.

The database server must be off-line because the older and newer versions share common files. You cannot install the database server if any of the common files are active.

WIN NT

The installation program automatically shuts down the old database server and starts the new database server. ♦

UNIX

Reconfigure the Operating System

You might need to change some of the kernel parameters for your UNIX operating system before you install the target database server. To reconfigure the operating system, follow the directions in the machine-notes file included on your target database server distribution media and the kernel-configuration instructions for your operating system.

WIN NT

Install the Database Server and Administration Tools

You can upgrade the administration tools only, the database server only, or both. If a previous version of the database server or administration tools is on the computer, the **Upgrade** page appears when you install the new product. The Installation wizard replaces the files but does not reconfigure the database server.

To perform the upgrade

1. When the **Upgrade** page appears, click **Next** for the INFORMIX-OnLine Installation wizard. Click **Next** again to begin the installation procedure.
2. Enter the serial number and serial-number key.

The installation program automatically verifies and brings down OnLine Workgroup Server, copies the new files, and preserves the database and dbspace data. Then the installation program starts OnLine Workgroup Server, Version 7.22, with the same configuration and shared-server computer.

The Installation wizard updates the SQLHOSTS registry keys and OSAHOSTS information automatically.

To install and configure onsnmp

Workgroup Replication and the **onsnmp** utility require Windows NT SNMP. The installation and upgrade program checks the registry for the SNMP master agent. If the master agent was not installed, the program displays a warning message but does not configure the registry for Workgroup Replication or **onsnmp**. If you later choose to install the SNMP master agent, you must run the **%INFORMIXDIR%\bin\inssnmp.exe** command-line utility to install the SNMP subagents. You need not reinstall the database server.

Install the Target Database Server

On UNIX platforms, you must be logged in as user **root,** or on Windows NT platforms, you must be a member of the **Administration** group to install your database server. Set the **INFORMIXDIR** environment variable to the directory where you plan to install your database server.

The installation script installs your database server into the **INFORMIXDIR** directory specified for user **root.** The installation script does not bring the database server on-line.

WIN NT

The setup program programs installs and brings up the database server on Windows NT. Follow the directions in your *Installation Guide* to install your database server. ♦

Warning: *If you install your database server in the same directory where the earlier version of the database server resides, the newer version overwrites the older files. If you want to preserve the files for the earlier version, you must install the newer database server in a different directory.*

Before you overwrite the older version, you must take the following precautions:

- If you do not have the original media for the older version, back up the **INFORMIXDIR** directory before you install your target database server.
- Copy the configuration file(s) in **INFORMIXDIR** in the **etc** directory to another location on the file system.

When you finish the installation and system reconfiguration, exit as user **root** and log on as user **informix**.

WIN NT

In Windows NT, you need to run the Installation wizard twice. First, upgrade the database server. Then install the administration tools.

The Installation wizard replaces the files but does not reconfigure the database server. If a previous version of the database server is on the computer, the **Upgrade** page appears when you install the new product.

The installation program automatically verifies and brings down your source database server, copies the new files, and preserves the database configuration information. The installation program starts your target database server with the same configuration and shared-server computer. ♦

Install Informix Enterprise Command Center

After you install the database server, install the administration tool, Informix Enterprise Command Center (IECC).

WIN NT

After you install the database server, IECC, or both, the **Informix Administration Tools** program group is available in your windowing environment. For complete information on installation, refer to the *Installation Guide for Informix Dynamic Server on Windows NT*. ♦

Customize the Database Server Environment

After you install your database server, ensure that the following environment variables are set to the correct values:

- **INFORMIXSERVER**
- **ONCONFIG**
- **PATH**
- **INFORMIXSQLHOSTS** (if used)

Important: *On UNIX, the client application looks for the* ***sqlhosts*** *file in the* ***etc*** *directory in the* ***INFORMIXDIR*** *directory. On Windows NT,* ***INFORMIXSQLHOSTS*** *points to the computer that contains the* ***sqlhosts*** *registry information. However, you can use the* ***INFORMIXSQLHOSTS*** *environment variable to change the location or name of the* ***sqlhosts*** *file.*

WIN NT

In Windows NT, the installation program sets the configuration parameters and environment variables for you. However, you can customize the configuration parameters in the **%ONCONFIG%** file and environment variables for your database server. In Windows NT, use **setnet32** to customize the environment variables on the client computer. ♦

For information about environment variables, refer to the *Informix Guide to SQL: Reference*.

UNIX

Update the ONCONFIG Configuration Parameters

You can customize your ONCONFIG configuration file and environment variables to take advantage of the new features that Informix Dynamic Server, Workgroup and Developer Editions, Version 7.3 introduces.

After you observe the performance of your database server, you might want to make further adjustments.

Important: *Use the same values for ROOTOFFSET, ROOTSIZE, and ROOTPATH that you used for the earlier version of your database server. Also use the same values for size and number of physical logs, logical logs, and for mirroring (if available).*

For information on how to configure, refer to your *Administrator's Guide*. For information about how to tune the configuration parameters, refer to your *Performance Guide*.

Update the Configuration Files

During the installation procedure, the install script checks the **etc** directory in the **INFORMIXDIR** directory for files named **config.arc**, **oper_deflt.arc, termcap, logevent.sh, sessalrm,** and **permalrm**. If these files do not exist, the install script provides them. If the files do exist, the install script does not overwrite the files. Instead, the install script provides additional files named **Config.arc**, **Oper_deflt.arc, Termcap, Logevent.sh, Sessalrm,** and **Permalrm** (note the initial uppercase letters).

Compare your current versions of the files with the new versions and determine whether new or changed configuration parameters or qualifiers exist.

WIN NT

Configure the Database Server to Use Enterprise Replication

Perform this step only if you plan to use Enterprise Replication or Workgroup Replication (Version 7.22 or later) with your database server.

Tip: *Informix Dynamic Server, Version 7.3, OnLine Workgroup Server, Version 7.2x, and OnLine Dynamic Server, Version 7.2x, on Windows NT only, can use Enterprise Replication.*

Before you can activate Enterprise Replication, you need to update the **$ONCONFIG** and **sqlhosts** files.

To activate Enterprise Replication for the first time

1. Bring your database server off-line.
2. Define one or more **dbserver** aliases for Enterprise Replication.
3. Define a group name for Enterprise Replication.
4. Bring your database server on-line.

5. For upgrades to version 7.22, verify that the SNMP service is installed on the host and that the SNMP subagents are registered.

   In Informix Dynamic Server, Workgroup and Developer Editions, Version 7.3, you need SNMP only if you plan to use the Enterprise Replication Monitoring Program.

6. Use Replication Manager to define each database server for replication.

   This step starts Enterprise Replication.

For complete information on how to configure for Enterprise Replication, refer to the *Guide to Informix Enterprise Replication*. For information on how to use SNMP, refer to the *Informix SNMP Subagent Guide*.

WIN NT

Installing and Configuring SNMP

In Informix Dynamic Server, Workgroup and Developer Editions, Version 7.3, if you plan to use the Enterprise Replication Monitoring Program, you must install SNMP. Enterprise Replication in Dynamic Server, Workgroup and Developer Editions, Version 7.3 does not require SNMP. The SNMP management tools use the **OnSnmpSubagent** to respond to queries. For database server Version 7.22, if you use Workgroup Replication on Windows NT, the **onsnmp** utility requires Windows NT SNMP. The installation and upgrade program checks the registry for the SNMP master agent. If the master agent is installed, the program displays a warning message but does not configure the registry for Workgroup Replication or **onsnmp**. If you later choose to install the SNMP master agent, you must run the **%INFORMIXDIR%\bin\inssnmp.exe** command line utility to install the SNMP subagents. You do not need to reinstall the database server.

Bring the Target Database Server On-Line

UNIX

When you bring your target database server on-line for the first time, bring it first to quiescent mode and then to on-line mode.

Execute the following command to bring your database server from off-line to quiescent mode:

```
oninit
```

Once the database server is in quiescent mode, check the message log for status messages.

Important: *If you note problems in the message file, solve the problems before you continue to the next step.*

Execute the following command to change your database server mode from quiescent mode to on-line mode:

```
onmode -m
```

The **sysmaster** and **sysutils** databases are created once your database server is brought on-line.

Warning: *The logical logs continue to fill with the transactions that result from the creation of the* ***sysmaster*** *database. If you run out of log space before the creation of the* ***sysmaster*** *database is complete, the database server halts with a "Logs Full" error. Thus, you must back up the logical logs.*

Use ON-Bar, ON-Archive, or **ontape** to back up logical logs. ♦

WIN NT

The setup program for Windows NT brings your database server on-line automatically.

If you customized the database server environment, bring down and restart the database server with the Informix Enterprise Command Center (IECC). When you restart your database server, the changes to the configuration parameters and environment variables take effect.

To start your database server with IECC

1. In the **IECC Console** program group, double-click the Informix Enterprise Command Center icon.
2. In the Informix Enterprise Command Center, select the database server in the tree view.
3. Choose **Server→On-line**.

You can start the database server with or without the Informix Enterprise Command Center interface. For more information, refer to *Informix Enterprise Command Center Installation Guide* or on-line help. ♦

UNIX

To start your database server with IECC

1. In the **INFORMIX-**OnLine Workgroup Server program group, double-click the **Command Center** icon. In the **Command Center**, choose a database server from the **Servers** menu.
2. On the **General** page, select **On-Line** in the **Server Mode** list box ♦

Run UPDATE STATISTICS

After you complete the migration procedure, run the UPDATE STATISTICS statement according to the recommended procedure in the *Informix Guide to SQL: Syntax*. UPDATE STATISTICS updates the information that your database server uses to plan efficient queries. For more information about UPDATE STATISTICS, refer to the *Informix Guide to SQL: Syntax*.

Verify the Integrity of the Data

Before you allow users to access the database, use the **oncheck** utility to verify that no data was corrupted in the migration process. You can verify the integrity of the reserve pages, system catalog tables, data, and indexes. For more information, see "Verify the Integrity of the Data" on page 7-8.

Make a Complete Backup of the Target Database Server

Use IECC to make a complete, whole-system backup of your database server. For more information, see the *Informix Enterprise Command Center Installation Guide* or the on-line help.

Use your database server backup utility (ON-Bar, ON-Archive, or **ontape)** to make a level-0 backup. Do not overwrite the tapes you used earlier when you made the final backup of your database server. For more information about how to make a backup, refer to the *Archive and Backup Guide for Informix Dynamic Server* or the *Backup and Restore Guide for Informix Dynamic Server*.

Important: *Do not restore the backed-up logical logs from the earlier version of your database server to the newer version of your database server.*

Verify the Access Path of Your SQL Statements

Use the SET EXPLAIN statement to verify that the access path of your SQL statements did not change when you migrated to OnLine Workgroup Server. If you have SET EXPLAIN output from the source database server, run SET EXPLAIN for OnLine Workgroup Server. Compare the SET EXPLAIN output from both the source and target database servers. SET EXPLAIN writes the access path that the optimizer chooses for each query to the SET EXPLAIN output file. The optimizer chooses the fastest path of execution for table joins.

WIN NT

For Windows NT, the SET EXPLAIN output filename is **%INFORMIXDIR%\sqlexpln\<*username*>.out**. ♦

UNIX

The UNIX SET EXPLAIN output filename is **sqexplain.out**. ♦

If the SET EXPLAIN output file shows that a different access path was used, complete the following steps:

1. Check the **OPTCOMPIND** environment variable or configuration parameter.
2. Check the **DBSPACETEMP** environment variable or configuration parameter to ensure that adequate temporary dbspaces are defined. You might need to define more temporary dbspaces.
3. Analyze the query access paths, and modify the schema to improve the performance if necessary.

Migration Complete

The first time your database server is brought on-line, the **sysmaster** database is built. Check the message log to ensure that the **sysmaster** database build is complete before you allow users to access the database server. After you complete a level-0 backup and you ensure that client users can access data on your database server, the migration process is complete.

Once you successfully migrate to Informix Dynamic Server, Workgroup and Developer Editions, Version 7.3, you might want to seek ways to obtain maximum performance. If you created sample queries for comparison, you can use them to characterize the performance differences between your earlier version and your new version. The results of these comparisons might suggest adjustments to configuration parameters, or to the layout of databases, tables, and chunks. For details on topics related to performance, refer to the *Performance Guide for Informix Dynamic Server*.

WIN NT

Changing Database Server Definitions

The installation program automatically updates the registry information and the database server definitions on the shared-server (SQLHOSTS) computer. The *database server definitions* consist of the database server name, TCP/IP host names, network types, and the service names.

Use Informix Enterprise Command Center if you want to change the **informix** user password, specify a different computer as the shared-server computer, or edit database server definitions (the database server name, TCP/IP host names, network types, and the service names). For example, you might specify a different shared-server computer if you have migrated the database server to a new computer or connected a single client to several database servers.

For information on how to use the **Client Control Panel** folder in Informix Enterprise Command Center to configure database servers, see the *Informix Enterprise Command Center User Guide* or the on-line help. For information on connectivity, see your *Administrator's Guide*.

GLS

Migrating to a GLS Locale

If you wish to migrate your database server to a non-English GLS locale, set the **DB_LOCALE** and **CLIENT_LOCALE** environment variables before you open the database in Version 7.3. If your previous database server version used Native Language Support (NLS), replace the NLS environment variables, such as **COLLCHAR**, with GLS environment variables. For information on how to work with locales and how to set GLS environment variables, see the *Informix Guide to GLS Functionality* and Chapter 10, "Changing Locales."

Reverting the Database Server and IECC

This section describes the steps for reverting from Informix Dynamic Server, Workgroup and Developer Editions, Version 7.3, and the Informix Enterprise Command Center 3.0 to an earlier version.

Follow the preparatory steps, described in "Preparing to Migrate Between Versions" on page 7-5, then complete the following steps:

WIN NT

1. Remove unsupported SQL features.
2. Uninstall Enterprise Replication Manager, if it is installed. ♦
3. Run the reversion utility (**onmode -b**).
4. Uninstall Informix Dynamic Server, Workgroup and Developer Editions and the IECC.
5. Remove GLS features (only if GLS was used and you are reverting to Version 7.12).
6. Modify configuration parameters.
7. Reset environment variables.
8. Reinstall the old version of your database server and IECC.
9. Bring the target database server on-line.
10. Verify the integrity of the data.
11. Make an initial, complete backup of your database server.

Remove Unsupported SQL Features

Before you revert, you must remove SQL features that the earlier version of your database server does not support. See the "New Features of This Product" section in the appropriate version of the *Informix Guide to SQL: Syntax*.

Uninstall Enterprise Replication, Version 7.3

Skip this section if Enterprise or Workgroup Replication is not installed on your system.

To revert to an earlier version if Enterprise Replication is active

1. Stop Enterprise Replication.

UNIX

2. For altered tables with CRCOLS, issue the command:

   ```
   alter table drop CRCOLS
   ```

   ♦

3. Execute the **onmode -b** command to revert to the earlier version of your database server.

Warning: *If you try to revert to a previous version of the database server while Enterprise Replication is active, the reversion will fail.*

To revert to an earlier version if Enterprise Replication is inactive

UNIX

1. In this situation, Enterprise Replication was previously active on this database server. For altered tables with CRCOLS, issue the command:

   ```
   alter table drop CRCOLS
   ```

   ♦

2. Execute the **onmode -b** command to revert to the earlier version of the database server. The **syscdr** database is dropped during reversion.

For more information, see the *Guide to Informix Enterprise Replication*.

Run the Reversion Utility

The database server must be running when you execute the reversion utility. Execute the reversion utility to revert the database, where *version_number* is the earlier database server version (see Figure 7-3 on page 7-23):

```
onmode -b version_number
```

After the reversion is complete, your database server is off-line. For more information about the **onmode -b** command, refer to Chapter 11, “Utilities for Data Migration.”

Revert From	Revert To	Command
Version 7.3	Version 7.22	**onmode -b 7.22**
Version 7.22	Version 7.2	**onmode -b 7.2**
Version 7.2x	Version 7.12	**onmode -b 7.1**

Figure 7-3
Reverting to an Earlier Database Server

Uninstall the Database Server and Informix Enterprise Command Center

Uninstall both the database server and the administration tool, Informix Enterprise Command Center.

WIN NT

To uninstall the product

1. Double-click the **Uninstall** icon in the **IECC Console** program group.
2. In the Uninstall dialog box, check **Remove OnLine Server**.

Warning: *Do not check* ***Remove all OnLine databases, supporting files and all database information****. If you check this option, your configuration, dbspaces, and database information will be lost, making reversion impossible.*

3. To uninstall the administration tools, check **Remove Administration Tools**.
4. Click **OK** to uninstall the database server.

For more information, see the *Informix Enterprise Command Center Installation Guide*.

GLS

Remove GLS Features (Only If Reverting to 7.12)

Skip this step if your database server uses the default English locale (**en_us.8859-1**). To revert the database server from GLS to Native Language Support (NLS) or Asian Language Support (ALS), set the appropriate NLS or ALS locales and environment variables. For information on how to work with locales, see the *Informix Guide to GLS Functionality* and Chapter 10, "Changing Locales."

Modify Configuration Parameters

Informix Dynamic Server, Workgroup and Developer Editions, Version 7.3 uses configuration parameters that did not appear in earlier versions of the database server that you should remove or modify prior to reversion.

For more information on these configuration parameters, see "Configuration Parameters Introduced in Version 7.3" on page 5-6.

Configuration Changes for Version 7.12

WIN NT

You must add the ONLANGMAP configuration parameter to your ONCONFIG file before you start OnLine Workgroup Server 7.12. ONLANGMAP takes the value **en_US-English**. For information on ONLANGMAP, see the INFORMIX-OnLine Dynamic Server for Windows NT, Version 7.12.TC2, documentation notes. ♦

If you are reverting to Version 7.12, remove the HETERO_COMMIT configuration parameter from your ONCONFIG file.

Configuration Changes for Version 7.22

You might need to revise the value of the ALARMPROGRAM configuration parameter in your configuration file.

If you used Enterprise Replication and are reverting to Version 7.12 or 7.2, remove the following parameters from your ONCONFIG file:

- CDR_LOGBUFFERS
- CDR_EVALTHREADS
- CDR_DSLOCKWAIT
- CDR_QUEUEMEM
- CDR_NIFUSEHELP
- CDR_NIFMEMS
- SCDR_NIFQUEUES

For more information on these configuration parameters, see the *Guide to Informix Enterprise Replication*.

Reset Environment Variables

Reset the environment variables to values that are appropriate for your version of your database server. For information on environment variables, refer to the *Informix Guide to SQL: Reference*.

NLS

Environment Variable Changes for Version 7.12

OnLine Workgroup Server, Version 7.12, supports NLS, not GLS. When you revert to Version 7.12, delete the following environment variables:

- **CC8BITLEVEL**
- **CLIENT_LOCALE**
- **DBCENTURY**
- **DBFLTMASK**
- **DBONPLOAD**
- **DB_LOCALE**
- **ESQLMF**
- **GLS8BITFSYS**
- **GL_DATES**
- **GL_DATETIME**
- **ONPLOAD**
- **PLCONFIG**
- **SERVER_LOCALE**
- **THREADLIB**

Add the following environment variables:

- **DBNLS**
- **COLLCHAR**
- **LANG**

Reinstall the Earlier Version of the Database Server

WIN NT

Reinstall the earlier version of the database server in the same directory as the files for the previous version.

To perform the installation

1. On the **Run Installation Again** page, select the **Copy all files, but leave configuration alone** installation option.
2. Supply your serial number and serial number key, as shown on the serial-number key card.
3. Select one or both of the components you want to install: **OnLine Database Server** or **Administration Tools**. Click **Next**.
4. The installation program automatically copies the older database server files and saves the configuration and the database data.

Warning: *Do not select* ***Copy all files and reconfigure the product****. If you select this option, your configuration and database information will be lost.*

For more information on installation, refer to the *INFORMIX-Enterprise Command Center User Guide*.

Warning: *If you are reverting to database server version 7.12, you must add the ONLANGMAP parameter to the ONCONFIG file before you start the reverted database server. ONLANGMAP takes the value* ***en_US-English*** *and ensures backward compatibility with non-NLS locates. If the database server fails on install, just restart it.* ♦

UNIX

You do not need to reinstall the administration tools because they were not upgraded for Version 7.2.

To perform the installation

1. Supply customer information.
2. On the **OnLine Workgroup Server Installation Options** screen, choose option 1, **Install**.
3. On the **Run Setup Again** screen, choose option 1, **Install All Files, But Leave Configuration Alone**. This option copies the new database server files without changing the configuration and **sqlhosts** file information.
4. The installation program automatically verifies and brings down the database server, copies the new files, and saves the database and dbspace data. It starts the earlier version of your database server with the old configuration.

Warning: *Do not select option 2,* ***Install OnLine Workgroup Server and Reconfigure****. If you select option 2, your configuration and database information will be lost.* ♦

Bring the Database Server On-Line

WIN NT

The installation program brings the database server on-line automatically. ♦

UNIX

Execute the following command to bring your database server to quiescent mode:

```
oninit
```

The database server initializes the shared memory and builds the **sysmaster** database. After the **sysmaster** database is built, the reversion process is complete. ♦

Verify the Integrity of the Data

To verify the integrity of your data, follow the steps as described in "Verify the Integrity of the Data" on page 7-8.

Back Up the OnLine Workgroup Server Data

Use Backup and Restore to make a complete backup of you database server. After you complete the reversion, Informix recommends that you make a level-0 backup. Use your preferred backup administration utility, ON-Bar, ON-Archive, or **ontape**, to make the backup. For information about how to make a backup, refer to your *Archive and Backup Guide* or see "Back Up Your Database Server" on page 7-9.

Important: *Do not overwrite the tapes that you used to back up your database server.*

Reversion Complete

After the **sysmaster** and **sysutils** databases are built, the reversion process is complete. Ensure that client users can access data on the earlier version of your database server.

Chapter

8

Migrating SE

This chapter describes how to migrate data between different versions of INFORMIX-SE database servers. The latest version of SE is Version 7.2x. Chapter 9, “Moving Between Database Servers,” covers moving data between SE and Informix Dynamic Server.

Importing and Exporting Data

You can use the following utilities to import and export data from an SE database server:

- **dbexport** / **dbimport**
- UNLOAD / **dbschema** / LOAD
- UNLOAD / **dbschema** / **dbload**

Let your target destination, performance and ease-of-use requirements determine the method that you use. Refer to Figure 1-9 on page 1-12 for details. Chapter 2, “Moving Between Computers,” describes the procedures for exporting and importing data. Chapter 11, “Utilities for Data Migration,” describes the syntax of the **dbexport**, **dbimport**, UNLOAD, **dbschema**, LOAD, and **dbload** utilities.

Migrating Between Different Versions of SE

Unlike Informix Dynamic Server data, SE data is stored in ordinary UNIX files. The structure of these files remained the same between versions of1 SE, so that migrating from one version to a later version requires little preparation. SE manages the file contents, but the operating system manages the I/O.

Preparing to Migrate from a Pre-6.0 SE to SE 7.2x

Version 6.0 of Informix products introduced changes in the way clients connect to database servers. The names of database server utilities also changed.

The sqlhosts File

Pre-6.0 versions of SE did not require an **sqlhosts** file unless you used SE with INFORMIX-NET. Beginning with Version 6.0, Informix products require an **sqlhosts** file to specify connections between clients and servers. For information about how to prepare your **sqlhosts** file, refer to your *Administrator's Guide*.

Environment Variables

Beginning with Version 6.0, SE requires the following environment variables:

- **INFORMIXDIR**
- **PATH**
- **INFORMIXSERVER**

Depending on your network configuration, you might also need the following environment variables:

- **SQLEXEC**
- **SQLRMDIR**
- **SQLRM**

For information about how to set these environment variables, refer to your *Administrator's Guide*.

SE Utilities

Beginning with Version 6.0, the names of the SE utilities **bcheck** and **dblog** changed to **secheck** and **selog**, respectively. If you have scripts that use these utilities, you must update the names of the utilities.

To migrate from a pre-6.0 version of SE to SE 7.2x

1. Ask all users to exit from their applications.

2. Verify the validity of your data.

For Version 4.1 or 5.x, execute the following command for each table in the database:

```
bcheck tablename
```

For more information about these utilities, refer to the appropriate version of the your *Administrator's Guide.*

3. Install SE, Version 7.2x.

For information about how to install SE, refer to the your *Installation Guide.*

Preparing to Migrate from a Post-6.0 SE to SE 7.2x

When you migrate from a post-6.0 version of SE to SE 7.2x, you do not need to change the environment variables or the **sqlhosts** file.

To migrate from a post-6.0 version of SE to SE 7.2x

1. Ask all users to exit from their applications.

2. Verify the validity of the data.

For Version 6.0 and later, execute the following command for each table in the database:

```
secheck tablename
```

For more information about these utilities, refer to the appropriate version of the *INFORMIX-SE Administrator's Guide.*

3. Install SE, Version 7.2x.

For information about how to install SE, refer to your *Installation Guide.*

Reverting to an Earlier Version of SE

Informix does not provide a reversion utility for SE. To move your database to an earlier version of SE, you must unload and then reload your data. Follow the procedures for **dbexport** and **dbimport** or UNLOAD that are described in "Moving Data to SE from Informix Dynamic Server or Its Editions" on page 9-11.

Converting C-ISAM Files to SE

This section describes how to convert C-ISAM files to SE format. C-ISAM files are organized differently than relational tables. C-ISAM files tend to be much larger and are not normalized. Therefore, you need to convert C-ISAM applications to the SQL format that you can use with SE:

1. In SE, use the SQL CREATE TABLE statement to create a table that corresponds to the data fields in the C-ISAM application. (Do not use the name of the C-ISAM file for the table name.)
2. Delete the empty**.dat** file that SE created in the CREATE TABLE statement. It has the name `tabname|tabid|".dat"`. Also, delete the empty**.idx** file.
3. Either move the C-ISAM file to the**.dbs** directory, or update the **systables** system catalog with the name of the C-ISAM data file.

   If you update both the C-ISAM and SE files with logging turned on, you must use common logging.

Migrating from C-ISAM to Dynamic Server or Its Editions

After you convert the C-ISAM data files to SE format, use **dbexport** and **dbimport** to migrate the data from SE to Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions. Follow the procedures that are described in"Moving Data to SE from Informix Dynamic Server or Its Editions" on page 9-11. For more information on how to work with C-ISAM files, see the *C-ISAM Programmer's Manual*.

Tip: *For details on how to convert* C-*ISAM data files to SE, contact your local systems engineer for assistance.*

Chapter

9

Moving Between Database Servers

This chapter describes the steps for how to migrate between Informix database servers. Figure 1-7 on page 1-10 shows the paths for migrating to a different environment. Figure 1-6 on page 1-9 shows the paths for migrating between different database servers in the same environment.

Choosing a Migration Method

UNIX and Windows NT store data in different page sizes. INFORMIX-SE stores data in a different format than does Informix Dynamic Server, Informix Dynamic Server with Universal Data Option, Informix Dynamic Server, Workgroup and Developer Editions, Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options.

When your migration involves different environments or SE, you must export data and its schema information from one database server and import the exported data into the other database server.

The method that you choose for exporting and importing data depends on how much data you plan to move. All of these methods deliver similar performance and allow you to modify the schema of the database. You can use the following migration methods:

- **dbexport** and **dbimport**

  To move an entire database, use the **dbexport** and **dbimport** utilities.
- UNLOAD, LOAD, and **dbschema**

  To move selected columns or tables, use the UNLOAD statement. Use LOAD when you do not wish to change the data format.
- UNLOAD, **dbload**, and **dbschema**

  To move selected columns or tables, use the UNLOAD statement. Use **dbload** to change the data format.

- **onunload** and **onload**

  To move data in page-sized chunks, use the **onunload** utility. Use the **onunload** utility to move data to an identical database server on a computer of the same type.

- High-Performance Loader (HPL)

  To move selected columns or tables or an entire database, use the HPL.

XPS

- External Tables

  To load and unload data in Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options, use the external tables. For more information on this topic, see Chapter 3, "Migrating OnLine Dynamic Server 7.2 to AD/XP 8.2." ♦

Important: *Do not use* ***onunload*** *and/or* ***onload*** *with Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options or Universal Data Option data; instead use external tables for Advanced Decision Support and Extended Parallel Options and the HPL for Universal Data Option.*

Moving Data from SE to Informix Dynamic Server or Its Editions

The following sections describe the steps for moving data from SE to Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions. You must choose a migration method, eliminate SE-specific features, migrate the data from SE, and modify client applications.

Using the dbexport and dbimport Utilities

If you intend to move an entire database from SE to Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions, the **dbexport** and **dbimport** combination is the easiest migration method.

To use dbexport and dbimport to move data from SE to Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions

1. Use the **dbexport** utility to export the data from SE.

   You can move the data to another directory or directly to tape.

Warning: *Do not use the **-ss** option with dbexport when you move data between database servers. The **-ss** option generates SE-specific syntax that Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions does not recognize.*

2. Edit the database schema file (the.**sql** file that **dbexport** creates).

   You might want to add information that Informix Dynamic Server (or Dynamic Server, Workgroup and Developer Editions) databases and tables can use, such as:

   - Initial- and next-extent values for a table
   - Lock mode for a table
   - Blobspace where TEXT or BYTE data types should reside
   - Dbspace where the tables should reside
   - VARCHAR or NVARCHAR column specifications
   - ODS Fragmentation schemes. ♦

3. You can also make the following changes to the schema file:

   - Alter ownership or SQL privileges on specific tables and indexes.
   - Specify the dbspace location of the database.
   - Change the logging status of the database.
   - Change the ANSI-compliance status of the database.
   - Remove unsupported SQL syntax, if necessary. (You can write a script to automate the task of modifying the SQL statements.)

   For details about supported SQL statements, see the *Informix Guide to SQL: Syntax*.

4. If necessary, install the new database server. For installation procedures, see your *Installation Guide*

Important: *For OnLine Dynamic Server, Version 7.1, or higher, you must access rows that were fragmented using column values (primary key values) in a row rather than ROWID.*

5. Change the **INFORMIXSERVER** environment variable to specify your target database server.
6. If necessary, use your standard file-transfer process to move the export files to the target computer. Ensure that both database servers use the same **DBDATE** and **DBMONEY** formats.
7. Start the new database server.
8. Use the **dbimport** utility to move the database data into the new database server.

For detailed descriptions of the **dbexport** and **dbimport** utilities, refer to Chapter 11, "Utilities for Data Migration."

Using the UNLOAD Statement

The UNLOAD statement lets you retrieve selected rows from a database and write those rows into a text file.

Using UNLOAD, LOAD, and dbschema

If you want to move selected tables or columns instead of an entire SE database to your target database server, use the SQL statements UNLOAD and LOAD in the DB-Access utility with the **dbschema** database utility.

To use UNLOAD, LOAD, and dbschema to move data from SE to Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions

1. Make sure that you have sufficient disk space to store the unloaded data. (You cannot unload data to tape with UNLOAD.)
2. Invoke the DB-Access utility.
3. Use the UNLOAD statement to move the selected data into a text file. Use a separate UNLOAD statement for each target table.
4. Exit from DB-Access.

5. If you plan to load data into a table or tables that do not exist, follow these steps:
   a. Use the **dbschema** utility to create a schema file from the SE database.
   b. Edit the schema file so that it describes the new tables.
   c. Make any of the following changes to the schema file:
      - Alter ownership or SQL privileges on specific tables and indexes.
      - Specify the dbspace location for the table.
      - Change the logging status of the database.
      - Change the ANSI-compliance status of the database.
      - Remove unsupported SQL syntax, if necessary.

   For details about supported SQL statements, see the *Informix Guide to SQL: Syntax*.

   If you prefer, you can omit this step and, in step 12, enter the statements that create the tables.
6. If necessary, install your target database server.
7. Change the **INFORMIXSERVER** environment variable to specify your target database server.
8. If necessary, use your standard file-transfer process to move the export files to the target computer. Ensure that both database servers use the same **DBDATE** and **DBMONEY** formats.
9. Start the new database server.
10. Invoke the DB-Access utility.
11. Select the target database.

    If you are creating a new database, execute the CREATE DATABASE statement, or choose **Database→Create** from the DB-Access menu.
12. If you plan to load data into a new table or tables, choose and run the schema file that you prepared in step 5, or enter CREATE TABLE statements to create the new tables.
13. If you created a new database, you might need to update the **DBPATH** environment variable with the new database location.
14. Use LOAD statements to load the data into the desired tables.

Using UNLOAD, dbload, and dbschema

If you need to manipulate the data before you load it into a new table, use a combination of the UNLOAD statement and the **dbschema** and **dbload** utilities.

To use UNLOAD, dbload, and dbschema to move data from SE to Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions

1. Follow steps 1 through 13 of "Using UNLOAD, LOAD, and dbschema" on page 9-8.
2. Build a command file to use with the **dbload** utility, which loads the data files into the tables.
3. Execute **dbload** to load the data as your command file directs.

For information on UNLOAD, LOAD, **dbload**, and **dbschema**, refer to Chapter 11, "Utilities for Data Migration." For information on how to use DB-Access, refer to the *DB-Access User Manual*.

Adapting Programs for Informix Dynamic Server or Its Editions

After you successfully move the SE data to Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions, verify that your application developers know the differences between both database servers. For more information about the server-specific limitations of SQL statements, refer to the *Informix Guide to SQL: Syntax* and the *Informix Guide to SQL: Reference*. For information about environment variables, see the *Informix Guide to SQL: Reference*.

For more information about how to configure your database server, refer to your *Administrator's Guide*.

SE

The following statements contain syntax and keywords that only SE recognizes:

- CHECK TABLE
- CREATE AUDIT
- DROP AUDIT
- RECOVER TABLE
- REPAIR TABLE
- ROLLFORWARD DATABASE
- START DATABASE ♦

Moving Data to SE from Informix Dynamic Server or Its Editions

When you move data to SE from Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions, you must choose a migration method, eliminate features that SE does not support, migrate the data, and modify applications.

Eliminating Features That SE Does Not Support

Before you export your source database server data to SE, you must eliminate or convert the following unsupported data types to a data type that SE supports:

- VARCHAR
- BYTE
- TEXT

Using the dbexport and dbimport Utilities

If you intend to move an entire database to SE, the **dbexport** and **dbimport** combination is the easiest migration method.

To use dbexport and dbimport to move data from your source database server to SE

1. Use **dbexport** to export the data from your source database server.

   You can move the data to a directory or directly to tape.

Warning: *Do not use the **-ss** option with **dbexport** when you move data between database servers. The **-ss** option generates syntax that is specific to Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions that SE does not recognize.*

2. Remove the following information from the CREATE TABLE statements in the schema file (the.**sql** file that **dbexport** creates):
   - Initial- and next-extent sizes
   - Lock modes
   - Dbspace names
   - Blobspace names
   - Logging modes
   - Table-fragmentation expressions
3. You can also make the following changes in the schema file:
   - Alter ownership or SQL privileges on specific tables and indexes.
   - Change the logging status of the database.
   - Change the ANSI-compliance status of the database.
4. If necessary, install SE. For installation instructions, refer to your *Installation Guide*.
5. Change the **INFORMIXSERVER** environment variable to specify your new database server.
6. If necessary, use your standard file-transfer process to move the export files to the target computer. Ensure that both database servers use the same **DBDATE** and **DBMONEY** formats.
7. Move to the directory where **dbimport** will store the SE database.
8. Use **dbimport** to move the data into an SE database.

For detailed descriptions of the **dbexport** and **dbimport** utilities, refer to Chapter 11, "Utilities for Data Migration."

Using the UNLOAD Statement

The UNLOAD statement lets you retrieve selected rows from a database and write those rows into a text file.

Using UNLOAD, LOAD, and dbschema

If you want to move selected tables or columns to SE instead of moving an entire database, use the UNLOAD and LOAD statements in the DB-Access utility with the **dbschema** utility.

To use UNLOAD, LOAD, and dbschema to move data to SE

1. Make sure that you have sufficient disk space to store the unloaded data. (The UNLOAD statement does not allow you to unload to tape.)
2. Invoke the DB-Access utility.
3. Use UNLOAD statements to move the selected data into text files. Use a separate UNLOAD statement for each target table.
4. Exit from DB-Access.
5. If you plan to load data into a table or tables that do not exist, follow these steps:

   a. Use the **dbschema** utility to create a schema file from the source database server.

   b. Edit the schema file so that it describes the new tables.

   If you prefer, you can omit this step and, in step 11, enter the statements that create the tables.
6. If necessary, install SE.
7. Change the **INFORMIXSERVER** environment variable to specify your new database server.
8. If necessary, use your standard file-transfer process to move the export files to the target computer. Ensure that both database servers use the same **DBDATE** and **DBMONEY** formats.
9. Invoke the DB-Access utility.

10. Select the target database.

    If you are creating a new database, execute the CREATE DATABASE statement, or choose **Database→Create** from the DB-Access menu.

11. If you plan to load data into a new table, choose and run the schema file that you prepared in step 5, or enter CREATE TABLE statements to create the new tables.
12. If you created a new database, you might need to update the **DBPATH** environment variable with the new database location.
13. Use LOAD statements to load the data into the desired tables.

Using UNLOAD, dbload, and dbschema

If you need to manipulate the data before you load it into a new table, use a combination of the UNLOAD statement and the **dbschema** and **dbload** utilities.

To use UNLOAD, dbload, and dbschema to move from Informix Dynamic Server or Dynamic Server, Workgroup and Developer Editions to SE

1. Follow steps 1 through 12 of "Using UNLOAD, LOAD, and dbschema" on page 9-13.
2. Build a command file to use with the **dbload** utility, which loads the data files into the tables.
3. Execute **dbload** to load the data as your command file directs.

For information on UNLOAD, LOAD, **dbload**, and **dbschema**, refer to Chapter 11, "Utilities for Data Migration." For information on how to use DB-Access, refer to the *DB-Access User Manual*.

Adapting Your Programs for SE

After you migrate data to SE, verify that your application developers know the differences between both database servers.

SE

SE does not use the ONCONFIG configuration file. SE supports a subset of the environment variables that Informix Dynamic Server and Dynamic Server, Workgroup and Developer Editions support. For more information about how to use SE, refer to the *INFORMIX-SE Administrator's Guide*. ♦

For more information about the differences between database servers and their interpretation of SQL, refer to the *Informix Guide to SQL: Reference*.

IDS

Only Informix Dynamic Server supports the following SQL statements:

- ALTER FRAGMENT
- GRANT FRAGMENT
- REVOKE FRAGMENT
- SET DATASKIP
- SET PDQPRIORITY ♦

W/D

- Only Informix Dynamic Server and Dynamic Server, Workgroup and Developer Editions support the following SQL statements:
- CREATE ROLE
- DROP ROLE
- RENAME DATABASE
- SET ISOLATION
- SET LOG
- SET ROLE
- SET SESSION AUTHORIZATION ♦

If you change the name or pathname of the database server, update the **DBPATH** environment variable with the location of the new database. For information about **DBPATH**, refer to the *Informix Guide to SQL: Reference*.

Moving Data from Informix Dynamic Server to Dynamic Server, Workgroup and Developer Editions

The following sections describe the steps for moving data from Informix Dynamic Server to Dynamic Server, Workgroup and Developer Editions in the same environment or different environments.

Moving Data in the Same Environment

Migration between Informix Dynamic Server and Dynamic Server, Workgroup and Developer Editions is automatic if they share the same environment, such as Windows NT. Automatic migration means you do not need to use a migration utility such as **dbexport** to move the data.

Complete all the migration steps except steps 4 and 10, described in the next sedtion.

Moving Data Between Different Environments

When you move data between UNIX and Windows NT, you must choose a migration utility, eliminate database server- and environment-specific features, migrate the data, and modify applications. Complete the following migration steps:

1. Save a copy of the current configuration files.
2. Verify the integrity of the data.
3. Make a final complete backup of your source database server.
4. If you are migrating to a different environment, export the source database servers.
5. Bring the source database sever off-line.
6. Install and configure the target database server.
7. Verify port numbers and **services** file.
8. Customize the database server environment. (This step is optional.)
9. Bring the target database server on-line.
10. Import the databases to the target database server.

11. Verify the integrity of the data.
12. Make an initial, complete backup of the target database server.
13. Run UPDATE STATISTICS.

Save a Copy of the Current Configuration Files

Save a copy of the current configuration files that you have modified. These should include:

- the current ONCONFIG file located in the **etc** subdistrict of your installation directory.
- the **sqlhosts** information.
- **adtcfg**, located in the **aaodir** subdistrict.
- **adtmasks.***, located in the **dbssodir** subdistrict.
- the ON-Archive configuration files located in the **etc** subdistrict.

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of the data. Figure 9-1 lists the commands for verifying data integrity.

Action	oncheck Command
Check reserve pages	`oncheck -cr`
Check extents	`oncheck -ce`
Check system catalog tables	`oncheck -cc` *database_name*
Check data	`oncheck -cD` *database_name*
Check indexes	`oncheck -cI` *database_name*

Figure 9-1
Commands for Verifying the Data Integrity

For information on **oncheck**, refer to your *Administrator's Guide*.

Back Up the Source Database Server

Use your preferred backup method to make a complete (level-0) backup of your database server.

WIN NT

The Windows NT environment does not support ON-Archive. ♦

The tape parameters must specify a valid tape device. Be sure to retain and properly label the tape volume that contains the backup. For more information on how to make backups, refer to the *Archive and Backup Guide for Informix Dynamic Server*.

Export the Source Database Server

If you are migrating to a different environment (for example, from UNIX to Windows NT), choose one of the following sets of migration utilities:

- **dbexport** and **dbimport** (see "Using the dbexport and dbimport Utilities" on page 9-23)
- UNLOAD, **dbschema**, and LOAD (see "Using UNLOAD, dbschema, and LOAD" on page 9-24)
- UNLOAD, **dbschema**, and **dbload** (see "Using UNLOAD, dbschema, and dbload" on page 9-26)

Skip this step if you are migrating to Dynamic Server, Workgroup and Developer Editions in the same environment.

Shut Down the Database Server

Communicate to client users how long you expect the database server to be off-line for the migration. Terminate all database server processes, and place your database server in quiescent mode (also called *administration mode*).

To shut down the database server gracefully

1. Warn all users that you plan to shut down the database server and wait for them to exit.
2. Become user **informix** on UNIX platforms, or on Windows NT, you must be a member of the **Informix-Admin** group.

   WIN NT

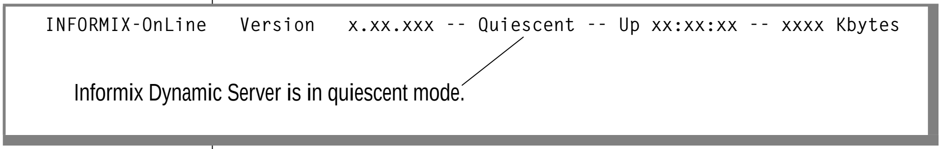
   Use **Services** in the Windows NT **Control Panel**. ♦

3. Execute the following command to take the database server to quiescent mode.

```
onmode -sy
```

4. Wait until your database server is in quiescent mode.

To verify the mode of your database server, execute the **onstat -** command. The first line of the **onstat** output contains the status of your database server. Figure 9-2 shows that the database server is in quiescent mode.



```
INFORMIX-OnLine   Version   x.xx.xxx -- Quiescent -- Up xx:xx:xx -- xxxx Kbytes
```

Informix Dynamic Server is in quiescent mode.

Figure 9-2
Example of onstat Status Line

5. Execute the following command to force a new logical log:

```
onmode -l
```

6. Execute the following command to force a checkpoint:

```
onmode -c
```

7. Execute the following command to shut down the database server:

```
onmode -yuk
```

Tip: *Monitor your log activity to verify that all commands were executed properly and to check for inconsistencies prior to migration.*

Install and Configure Dynamic Server, Workgroup and Developer Editions

If you have not already done so, follow the instructions in your *Installation Guide* to install and configure your target database server.

WIN NT

You can install Dynamic Server, Workgroup and Developer Editions and the Informix Enterprise Command Center on either the same or different computers. ♦

UNIX

You can install the Informix Enterprise Command Center on a personal computer that runs Windows 95 or Windows NT, and the database server on a UNIX computer. The installation program also starts the server agent, which is the communication link between Dynamic Server, Workgroup and Developer Editions and the IECC client. ♦

Use the **Setup** program to specify the network protocol and the computer on which Dynamic Server, Workgroup and Developer Editions looks for the database server definitions (**sqlhosts** and **osahosts** definitions).

Verify Port Numbers and Services File

The **services** file contains service names, port numbers, and protocol information. If you have installed the database server and the administration tools on different computers, verify that the port number listed in the **services** file is the same on the client and on the server computers.

WIN NT

The **services** file resides in the **\windir\system32\drivers\etc** directory. ♦

UNIX

For UNIX operating systems not running NIS, the **services** file resides in the **/etc/services** directory on the server and in the **\windir\services** directory on the Windows 95 client. ♦

Customize the Database Server Environment

If you are an advanced user, you can customize the ONCONFIG configuration file and environment variables for Dynamic Server, Workgroup and Developer Editions. Use a text editor to edit the ONCONFIG file. For more information on configuration parameters, refer to your *Installation Guide*.

Important: *Use the same values for ROOTOFFSET, ROOTSIZE, and ROOTPATH that you used for your old database server.*

You may want to customize new environment variables on the client. For more information on environment variables, refer to the *Informix Guide to SQL: Reference*.

Bring Dynamic Server, Workgroup and Developer Editions On-Line

The installation program brings the target database server on-line automatically.

If you customized the database server environment, bring down and restart your database server with the IECC. When you restart Dynamic Server, Workgroup and Developer Editions, the changes to the configuration parameters and environment variables take effect.

WIN NT

To start the database server with the IECC

1. In the **Informix Administration Tools** program group, double-click the **Command Center** icon.
2. In the IECC, select the database server in the **All Servers** tree view or the **Servers** list box.
3. Choose **Server→On-line**.

For more information, refer to the *Informix Enterprise Command Center User Guide* or the on-line help. ♦

UNIX

To start your database server at the UNIX command line, enter `oninit` on the server computer. ♦

Import the Databases to the Target Database Server

To load the databases into Dynamic Server, Workgroup and Developer Editions, use **dbimport**, LOAD, or **dbload,** depending on which utility you used to export the databases.

Verify the Integrity of the Data

To verify the integrity of data, use the **oncheck** utility, as described in "Verify the Integrity of the Data" on page 9-17.

Back Up the Target Database Server

To make a complete, whole-system backup of your target database server, use the Backup and Restore tool. For more information on backing up, refer to the *Informix Enterprise Command Center User Guide* or the on-line help.

WIN NT

Double-click the **Backup and Restore** icon in the **Informix Administration Tools** program group. ♦

UNIX

To launch Backup and Restore, enter **$INFORMIXDIR/bin/bar** on the UNIX computer where the database server is installed. ♦

Run UPDATE STATISTICS

After you complete the migration procedure, run the UPDATE STATISTICS statement on the server. The UPDATE STATISTICS statement updates the information that OnLine Workgroup Server uses to plan efficient queries. For more information about UPDATE STATISTICS, refer to the *Informix Guide to SQL: Syntax*.

Migration Complete

The first time your target database server is brought on-line, the **sysmaster** and the **sysutils** databases are built. Check the message log to ensure that the **sysutils** and **sysmaster** databases are created successfully before you allow users to access the database server. After you ensure that client users can access data on your target database server, the migration process is complete.

Adapting Your Programs for Dynamic Server, Workgroup and Developer Editions

After you successfully migrate the database server data, verify that your application developers know the differences between the source and target database servers. Dynamic Server, Workgroup and Developer Editions supports the same features as Informix Dynamic Server, except the following features:

- Fragmentation (also known as partitioning)
- High-Performance Loader (HPL)
- Parallel data query (PDQ)
- Role separation

For information on the SQL statements that Dynamic Server, Workgroup and Developer Editions supports, refer to the *Informix Guide to SQL: Syntax* manual and the *Informix Guide to SQL: Reference*.

Using the dbexport and dbimport Utilities

If you intend to move an entire database from Informix Dynamic Server to Dynamic Server, Workgroup and Developer Editions in different environments, the **dbexport** and **dbimport** combination is the easiest migration method:

1. Use **dbexport** to export the data from Informix Dynamic Server.

   You can move the data to a directory or directly to tape. Do not use the **-ss** option when you move data between database servers.

W/D

2. Remove table-fragmentation expressions from the CREATE TABLE statements in the schema file (the.**sql** file that **dbexport** creates).

   If tables are fragmented, **dbimport** might not work. Informix recommends that you defragment the tables before you export the database. ♦

3. You can also make the following changes in the schema file:
   - Alter ownership or SQL privileges for specific tables and indexes.
   - Change the logging status of the database.
   - Change the ANSI-compliance status of the database.
   - Remove unsupported SQL syntax, if necessary.
4. Follow the instructions in your *Installation Guide* and your *Administrator's Guide* to install and configure Dynamic Server, Workgroup and Developer Editions.
5. Change the **INFORMIXSERVER** environment variable to specify your new database server.
6. If necessary, use your standard file-transfer process to move the export files to the target computer. Ensure that both database servers use the same **DBDATE** and **DBMONEY** formats.
7. Move to the directory where **dbimport** will store the Dynamic Server, Workgroup and Developer Editions database.
8. Use **dbimport** to move the data to a Dynamic Server, Workgroup and Developer Editions database.

For detailed descriptions of the **dbexport** and **dbimport** utilities, refer to Chapter 11, "Utilities for Data Migration."

Using the UNLOAD Statement

The UNLOAD statement lets you retrieve selected rows from a database and write those rows to a text file.

Using UNLOAD, dbschema, and LOAD

If you want to move selected tables or columns instead of an entire Informix Dynamic Server database to Dynamic Server, Workgroup and Developer Editions, use the UNLOAD and LOAD statements in the DB-Access utility with the **dbschema** utility.

To use UNLOAD, dbschema, and LOAD to move data from Informix Dynamic Server to Dynamic Server, Workgroup and Developer Editions

1. Make sure that you have sufficient disk space to store the unloaded data. (The UNLOAD statement does not allow you to unload to tape.)
2. Invoke the DB-Access utility.
3. To move the selected data into text files, use UNLOAD statements. Use a separate UNLOAD statement for each target table.
4. Exit from DB-Access.
5. If you plan to load data into a table or tables that do not exist, follow these steps:
   a. To create a schema file from the source database server, use the **dbschema** utility.
   b. Edit the schema file so that it describes the new tables.

      If you prefer, you can omit this step and, in step 12, enter the statements that create the tables.
6. Follow the instructions in your *Installation Guide* and your *Administrator's Guide* to install and configure Dynamic Server, Workgroup and Developer Editions.
7. Change the **INFORMIXSERVER** environment variable to specify your new database server.
8. If necessary, use your standard file-transfer process to move the export files to the target computer. Ensure that both database servers use the same **DBDATE** and **DBMONEY** formats.
9. Invoke the DB-Access utility.
10. Select the target database.
11. If you are creating a new database, execute the CREATE DATABASE statement, or choose **Database→Create** from the DB-Access menu.
12. If you plan to load data into a new table, choose and run the schema file that you prepared in step 5, or enter CREATE TABLE statements to create the new tables.
13. If you created a new database, you might need to update the **DBPATH** environment variable with the new database location.
14. To load the data into the desired tables, use LOAD statements.

Using UNLOAD, dbschema, and dbload

If you need to manipulate the data in the specified UNLOAD file before you load it into a new table, use a combination of the UNLOAD statement and the **dbschema** and **dbload** utilities.

To use UNLOAD, dbschema, and dbload to move from Informix Dynamic Server to Dynamic Server, Workgroup and Developer Editions

1. Follow steps 1 through 13 from "Using UNLOAD, dbschema, and LOAD" on page 9-24.
2. Build a command file to use with the **dbload** utility, which loads the data files into the tables.
3. Execute **dbload** to load the data as your command file directs.

For information on UNLOAD, LOAD, **dbload**, and **dbschema**, refer to Chapter 11, "Utilities for Data Migration." For information on how to use DB-Access, refer to the *DB-Access User Manual*.

Moving Data from Dynamic Server, Workgroup and Developer Editions to Informix Dynamic Server

The following sections describe the steps for moving data from Dynamic Server, Workgroup and Developer Editions to Informix Dynamic Server in the same environment or different environments.

Moving Data in the Same Environment

Migration between Dynamic Server, Workgroup and Developer Editions and Informix Dynamic Server is automatic if they share the same environment, such as Windows NT or UNIX. Automatic migration means you do not need to use a migration utility such as **dbexport** to move the data.

Complete the following migration steps, described in detail on pages 9-28 through 9-32:

1. Save a copy of the current configuration files.
2. Verify the integrity of the data.

3. Make a final complete backup of your source database server.
4. Bring your source database server off-line.
5. Install the target database server.
6. Customize the database server environment. (This step is optional.)
7. Bring the target database server on-line.
8. Verify the integrity of the data.
9. Make an initial, complete backup of the target database server.
10. Run UPDATE STATISTICS.

Important: *If you want to install on Windows NT, select the **Copy all files, but leave configuration alone** option on the **Run Installation Again** page. The installation program automatically copies the new database server files, and saves the configuration and the database data.*

*Do not select the **Copy all files and reconfigure the product** option, or your configuration and database information will be lost.*

After you ensure that client users can access data on the target database server, the migration process is complete.

Moving Data Between Different Environments

When you move data between UNIX and Windows NT, you must choose a migration utility, eliminate database server- and environment-specific features, migrate the data, and modify applications.

Complete the following migration steps:

1. Save a copy of the current configuration files.
2. Verify the integrity of the data.
3. Make a final complete backup of the source database server.
4. If you are migrating to a different environment, export the source database server.
5. Bring the source database server off-line.
6. Install and configure the target database server and the administration tools.
7. Verify port numbers and **services** file.

8. Customize the database server environment. (This step is optional.)
9. Bring the target database server on-line.
10. Import the databases into the target database server.
11. Verify the integrity of the data.
12. Make an initial, complete backup of the target database server.
13. Run UPDATE STATISTICS.

Save a Copy of the Current Configuration Files

Save a copy of the current configuration files that you have modified. These should include:

- the current ONCONFIG file located in the **etc** subdistrict of your installation directory.
- the **sqlhosts** information.
- **adtcfg**, located in the **aaodir** subdistrict.
- **adtmasks.***, located in the **dbssodir** subdistrict.
- the ON-Archive configuration files located in the **etc** subdistrict.

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of data, as described in "Verify the Integrity of the Data" on page 9-17.

Back Up the Source Database Server

Use the Backup and Restore tool to make a complete, whole-system backup of the source database server. For more information on how to back up, refer to the *INFORMIX-Enterprise Command Center User Guide* or the on-line help.

WIN NT

Double-click the **Backup and Restore** icon in the **Informix Administration Tools** program group. ♦

UNIX

To launch Backup and Restore, enter **$INFORMIXDIR/bin/bar** on the UNIX computer where the database server is installed. ♦

Export the Source Database Server

If you are migrating to a different environment (for example, from UNIX to Windows NT), choose one of the following sets of migration utilities:

- **dbexport** and **dbimport** (see "Using the dbexport and dbimport Utilities" on page 9-33)
- UNLOAD, **dbschema**, and LOAD (see "Using UNLOAD, LOAD, and dbschema" on page 9-34)
- UNLOAD, **dbschema**, and **dbload** (see "Using UNLOAD, dbschema, and dbload" on page 9-35)

Skip this step if you are migrating to Informix Dynamic Server in the same environment.

Shut Down the Source Database Server

UNIX

Select **Off-Line** from the **Server Mode** list box in the **General** tab of the Informix Enterprise Command Center. ♦

WIN NT

In the Informix Enterprise Command Center, select the database server in the **All Servers** tree view or the **Servers** list box. Choose **Server→Off-line**. ♦

Install the Target Database Server

Install and configure the target database server according to the instructions in your *Installation Guide*.

UNIX

You must be user **root** to install the product. When you finish the installation and system reconfiguration, exit as user **root** and log in as user **informix**. ♦

WIN NT

You can install the target database server and the administration tools on either the same or different computers.

The installation program automatically starts the target database server.

Use **Setup** to specify the network protocol and the computer on which the target database server looks for the database server definitions. ♦

Verify Port Numbers and Services File

The **services** file contains service names, port numbers, and protocol information. If you have installed the database server and the administration tools on different computers, verify that the port number listed in the **services** file is the same on the client and the server computers.

WIN NT

The **services** file resides in the **\windir\system32\drivers\etc** directory. ♦

UNIX

The **services** file resides in the **/etc/services** directory on the server and in the **\windir\services** directory on the Windows NT client. ♦

Customize the Database Server Environment

If you are an advanced user, you can customize the ONCONFIG configuration file and environment variables for OnLine Dynamic Server. To edit the ONCONFIG file, use a text editor. For more information on configuration parameters, refer to your *Administrator's Guide*. For more information on environment variables, refer to the *Informix Guide to SQL: Reference*.

Important: *Use the same values for ROOTOFFSET, ROOTSIZE, and ROOTPATH that you used for your old database server.*

Bring the Target Database Server On-Line

The installation program brings the target database server on-line automatically.

If you customized the database server environment, bring down and restart the target database server with the Informix Enterprise Command Center. When you restart the target database server, the changes to the configuration parameters and environment variables take effect.

WIN NT

To start Informix Dynamic Server on Windows NT

1. In the **Informix Administration Tools** program group, double-click the Informix Enterprise Command Center icon.
2. In the Informix Enterprise Command Center, select the database server in the **All Servers** tree view or the **Servers** list box.
3. Choose **Server→On-line**.

For more information, refer to the *Informix Enterprise Command Center User Guide* or the on-line help. ♦

UNIX

To start Informix Dynamic Server on UNIX

1. Enter **oninit** at the command-line prompt.
2. If you wish to initialize the database server, enter **oninit -i**. ♦

Import the Databases into the Target Database Server

Use **dbimport**, LOAD, or **dbload** to load the databases into Informix Dynamic Server, depending on which utility you used to export the databases.

Verify the Integrity of the Data

Use the **oncheck** utility to verify the integrity of data, as described in "Verify the Integrity of the Data" on page 9-17.

Back Up the Target Database Server

WIN NT

Use Backup and Restore in the **Command Center** to make a complete, whole-system backup of the target database server. For more information, refer to *Informix Enterprise Command Center User Guide* or the on-line help. ♦

UNIX

Use ON-Bar, ON-Archive, or **ontape** to make a complete, whole-system backup of Informix Dynamic Server. For more information, refer to the *Archive and Backup Guide for Informix Dynamic Server* or the *Backup and Restore Guide for Informix Dynamic Server*. ♦

Run UPDATE STATISTICS

After you complete the migration procedure, run the UPDATE STATISTICS statement on the server. The UPDATE STATISTICS statement updates the information that OnLine Dynamic Server uses to plan efficient queries. For more information about UPDATE STATISTICS, refer to the *Informix Guide to SQL: Syntax*.

Migration Complete

The first time the target database server is brought on-line, the **sysmaster** and **sysutils** databases are built. Check the message log to ensure that the database builds are complete before you allow users to access the database server. After you ensure that client users can access data on your target database server, the migration process is complete.

Adapting Your Programs for Informix Dynamic Server

After you successfully move the Dynamic Server, Workgroup and Developer Editions data to Informix Dynamic Server, verify that your application developers know the differences between both database servers. Informix Dynamic Server supports the same features as Dynamic Server, Workgroup and Developer Editions, plus the following features:

- Fragmentation (also known as partitioning)
- High-Performance Loader (HPL)
- Parallel data query (PDQ)
- Role separation

For information on the SQL statements that Informix Dynamic Server supports, refer to the *Informix Guide to SQL: Syntax* manual and the *Informix Guide to SQL: Reference.*

Using the dbexport and dbimport Utilities

If you intend to move an entire database from Dynamic Server, Workgroup and Developer Editions to Informix Dynamic Server in different environments, the **dbexport** and **dbimport** combination is the easiest migration method:

1. Use **dbexport** to export the data from the old database server.

   You can move the data to a directory or directly to tape. Do not use the **-ss** option when you move data between database servers.

UNIX

2. You might want to add the following information that Informix Dynamic Server databases and tables can use:
   - Fragmentation schemes
   - PDQ support ♦
3. You can also make the following changes in the schema file:
   - Alter ownership or SQL privileges for specific tables and indexes.
   - Change the logging status of the database.
   - Change the ANSI-compliance status of the database.
   - Remove unsupported SQL syntax, if necessary.
4. Follow the instructions to install and configure Informix Dynamic Server.
5. Change the **INFORMIXSERVER** environment variable to specify your new database server.
6. If necessary, use your standard file-transfer process to move the export files to the target computer. Ensure that both database servers use the same **DBDATE** and **DBMONEY** formats.
7. Move to the directory where **dbimport** will store the Informix Dynamic Server database.
8. Use **dbimport** to move the data to an Informix Dynamic Server database.

For detailed descriptions of the **dbexport** and **dbimport** utilities, refer to Chapter 11, "Utilities for Data Migration."

Using the UNLOAD Statement

The UNLOAD statement lets you retrieve selected rows from a database and write those rows to a text file.

Using UNLOAD, LOAD, and dbschema

If you want to move selected tables or columns instead of an entire Dynamic Server, Workgroup and Developer Editions database to Informix Dynamic Server, use the UNLOAD and LOAD statements in the DB-Access utility with the **dbschema** utility.

To use UNLOAD, LOAD, and dbschema to move data from Dynamic Server, Workgroup and Developer Editions to Informix Dynamic Server

1. Make sure that you have sufficient disk space to store the unloaded data. (The UNLOAD statement does not allow you to unload to tape.)
2. Invoke the DB-Access utility.
3. Use UNLOAD statements to move the selected data into text files. Use a separate UNLOAD statement for each target table.
4. Exit from DB-Access.
5. If you plan to load data into a table or tables that do not exist, follow these steps:
   a. Use the **dbschema** utility to create a schema file from the Informix Dynamic Server database.
   b. Edit the schema file so that it describes the new tables.

   If you prefer, you can omit this step and, in step 12, enter the statements that create the tables.
6. Follow the instructions to install and configure Informix Dynamic Server.
7. Change the **INFORMIXSERVER** environment variable to specify your new database server.
8. If necessary, use your standard file-transfer process to move the export files to the target computer. Ensure that both database servers use the same **DBDATE** and **DBMONEY** formats.

9. Invoke the DB-Access utility.
10. Select the target database.
11. If you are creating a new database, execute the CREATE DATABASE statement, or choose **Database→Create** from the DB-Access menu.
12. If you plan to load data into a new table, choose and run the schema file that you prepared in step 5, or enter CREATE TABLE statements to create the new tables.
13. If you created a new database, you might need to update the **DBPATH** environment variable with the new database location.
14. Use LOAD statements to load the data into the desired tables.

Using UNLOAD, dbschema, and dbload

If you need to manipulate the data in the specified UNLOAD file before you load it into a new table, use a combination of the UNLOAD statement and the **dbschema** and **dbload** utilities.

To use UNLOAD, dbschema, and dbload to move from Dynamic Server, Workgroup and Developer Editions to Informix Dynamic Server

1. Follow steps 1 through 13 from "Using UNLOAD, LOAD, and dbschema" on page 9-34.
2. Build a command file to use with the **dbload** utility, which loads the data files into the tables.
3. Execute **dbload** to load the data as your command file directs.

For information on UNLOAD, LOAD, **dbload**, and **dbschema**, refer to Chapter 11, "Utilities for Data Migration." For information on how to use DB-Access, refer to the *DB-Access User Manual*.

Moving Informix Dynamic Server Data Between Environments

The following section describes the steps for moving Informix Dynamic Server, Version 7.2x, data between UNIX and Windows NT environments.

Procedure for Moving Data Between Environments

Complete the following migration steps:

1. Save a copy of the current configuration files. For detailed steps, see "Save a Copy of the Current Configuration Files" on page 9-17.
2. Use ON-Bar, ON-Archive, or **ontape** to make a final level-0 backup.

   For more information, refer to the *Archive and Backup Guide for Informix Dynamic Server* or the *Backup and Restore Guide for Informix Dynamic Server*.
3. Choose one of the following sets of migration utilities to unload the databases:
   - **dbexport** and **dbimport**
   - UNLOAD, **dbschema**, and LOAD
   - UNLOAD, **dbschema**, and **dbload**
4. Bring Informix Dynamic Server off-line.
5. Install and configure the new version of Informix Dynamic Server. If you are migrating to Windows NT, also install the administration tools.
6. Bring Informix Dynamic Server on-line.
7. Use **dbimport**, LOAD, or **dbload** to load the databases into Informix Dynamic Server, depending on which utility you used to export the databases.
8. Make an initial level-0 backup under Informix Dynamic Server.
9. Run UPDATE STATISTICS to update the information that the server uses to plan efficient queries.

Using the Migration Utilities

Choose one of the following migration utilities:

- If you intend to move an entire database on Informix Dynamic Server between different environments, the **dbexport** and **dbimport** combination is the easiest migration method. Follow the steps in "Using the dbexport and dbimport Utilities" on page 9-33.
- If you want to move selected tables or columns, instead of an entire database, use the UNLOAD and LOAD statements with the **dbschema** utility. Follow the steps in "Using the UNLOAD Statement" on page 9-34.
- If you need to manipulate the data in the specified UNLOAD file before you load it into a new table, use a combination of the UNLOAD statement and the **dbschema** and **dbload** utilities. Follow the steps in "Using UNLOAD, dbschema, and dbload" on page 9-35.

Adapting Your Programs for the UNIX or Windows NT Environment

Certain Informix Dynamic Server configuration parameters and environment variables are environment dependent. For details, see your *Administrator's Guide* and Appendix A, "Database Server Environment Variables."

Informix Dynamic Server, Version 7.3, supports Enterprise Replication and the Informix Enterprise Command Center.

Migration Complete

The first time Informix Dynamic Server is brought on-line, the **sysmaster** and **sysutils** databases are built. Check the message log to ensure that the **sysmaster** and **sysutils** databases have been created successfully before you allow users to access the database server. After you ensure that client users can access data on Informix Dynamic Server, the migration process is complete. Then you might want to seek ways to obtain maximum performance. For details on topics related to performance, refer to your *Performance Guide*.

Moving Dynamic Server, Workgroup and Developer Editions Data Between Environments

The following section describes the steps for moving Dynamic Server, Workgroup and Developer Editions data between UNIX and Windows NT environments.

Procedure for Moving Data Between Environments

Complete the following migration steps:

1. Save a copy of the current configuration files. For detailed steps, see "Save a Copy of the Current Configuration Files" on page 9-28.
2. UNIX
   Use **ontape**, ON-Bar, or the Backup and Restore tool to make a complete, whole-system backup. ♦

   WIN NT
   For more information about making backups, refer to *Archive and Backup Guide for Informix Dynamic Server* and *Backup and Restore Guide for Informix Dynamic Server*. ♦
3. Choose one of the following sets of migration utilities to unload the databases:
   - **dbexport** and **dbimport**
   - UNLOAD, LOAD, and **dbschema**,
   - UNLOAD, **dbload**, and **dbschema**
4. Bring Dynamic Server, Workgroup and Developer Editions off-line.
5. WIN NT
   Install and configure the new version of Dynamic Server, Workgroup and Developer Editions. If you are migrating to Windows NT, also upgrade the administration tools.

   The installation program brings Dynamic Server, Workgroup and Developer Editions on-line automatically.
6. Use **dbimport**, LOAD, or **dbload** to load the databases into Dynamic Server, Workgroup and Developer Editions, depending on which utility you used to export the databases.

7. Make a complete, whole-system backup of Dynamic Server, Workgroup and Developer Editions.
8. Run UPDATE STATISTICS to update the information that the server uses to plan efficient queries.

Using the Migration Utilities

Choose one of the following migration utilities:

- If you intend to move an entire database on Dynamic Server, Workgroup and Developer Editions between different environments, the **dbexport** and **dbimport** combination is the easiest migration method. Follow the steps in "Using the dbexport and dbimport Utilities" on page 9-23.
- If you want to move selected tables or columns, instead of an entire database, use the UNLOAD and LOAD statements with the **dbschema** utility. Follow the steps in "Using the UNLOAD Statement" on page 9-24.
- If you need to manipulate the data in the specified UNLOAD file before you load it into a new table, use a combination of the UNLOAD statement and the **dbschema** and **dbload** utilities. Follow the steps in "Using UNLOAD, dbschema, and dbload" on page 9-26.

Adapting Your Programs for the UNIX or Windows NT Environment

UNIX

Dynamic Server, Workgroup and Developer Editions for UNIX is at Version 7.2. It supports GLS and uses Version 1.0 of the INFORMIX-Enterprise Command Center. ♦

WIN NT

Dynamic Server, Workgroup and Developer Editions for Windows NT is at Version 7.3. It supports GLS, ON-Bar, Enterprise Replication, the Gateway products, and uses Version 3.0 of the Informix Enterprise Command Center. ♦

Certain Dynamic Server, Workgroup and Developer Editions configuration parameters and environment variables are environment dependent. For details, see your *Administrator's Guide* and Appendix A, "Database Server Environment Variables."

Migration Complete

The first time Dynamic Server, Workgroup and Developer Editions is brought on-line, the **sysmaster** and **sysutils** databases are built. Check the message log to ensure that the build for these databases is complete before you allow users to access the database server. After you ensure that client users can access data on Dynamic Server, Workgroup and Developer Editions, the migration process is complete.

Chapter

Changing Locales

10

Informix products use the following language-support features:

ALS

- Informix Asian Language Support (ALS) provides support for Asian (multibyte) characters. ♦

NLS

- Informix Native Language Support (NLS) provides support for single-byte, non-English characters.♦

GLS

- Informix Global Language Support (GLS) provides support for both multibyte and single-byte characters. ♦

This chapter describes how to migrate a database that uses the NLS or ALS feature for its language support to the new GLS support. The chapter discusses the following topics:

- Understanding Informix language support
- Understanding environment variables used for language support
- Converting an NLS or ALS database to work with the GLS feature
- Reverting a GLS database to work with the NLS or ALS feature

For an introduction to the GLS feature, see the *Informix Guide to GLS Functionality*.

Understanding Informix Language Support

By default, most Informix products assume that a database contains character data that follows the conventions of U.S. English. These conventions include the following:

- Character data that uses a code set that supports U.S. English characters (the ASCII code set)
- Classification of character data into English categories
- Collation of character data in ASCII code-set order

A *non-English database* is a database whose character data is in some language other than English. Informix has provided the ALS language support for non-English databases with Asian (multibyte) characters and NLS language support for non-English databases with single-byte characters. With Version 7.2, Informix created a single feature to provide support for single-byte and multibyte data in non-English languages. This feature is called Informix Global Language Support (GLS). The following sections summarize each of the Informix language support features.

GLS

Global Language Support (GLS)

The GLS feature provides support for both single-byte and multibyte non-English data in a database application. This feature provides the following advantages to a database application:

- Truly internationalized support for languages

  The database server that uses the GLS feature can support data in both single-byte and multibyte code sets. You do not need to install two different products.
- Informix-specific locale files

  The *GLS locale* is not operating-system dependent. Your database, database server, and client applications are no longer dependent on the language support that your operating system provides.

This section provides the following information about the GLS feature:

- Which Informix products support the GLS feature?
- What is a GLS locale?
- What are the GLS environment variables?
- How do client applications perform code-set conversion with a GLS database server?

For a more complete description of the GLS feature, see the *Informix Guide to GLS Functionality*.

Informix GLS Products

Informix GLS products are those that use the GLS feature to provide support for non-English database applications. Informix supports the GLS feature in the following types of products:

- Database server products
- Client products

Informix introduced GLS in its Version 7.2 database servers.

GLS Database Servers

A database server that uses the GLS feature for its language support is called a *GLS database server.* A database server uses the GLS feature to correctly manipulate character, numeric, and date quantities, as well as to sort character data. As Figure 10-8 on page 10-21 shows, the following database servers are GLS database servers:

- Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options, Version 8.2, and later
- INFORMIX-Universal Server and Informix Dynamic Server with Universal Data Option
- OnLine Dynamic Server, Versions 7.2, through Informix Dynamic Server, 7.3
- INFORMIX-OnLine Workgroup Server, Versions 7.2, through Informix Dynamic Server, Workgroup and Developer Editions, 7.3
- INFORMIX-SE, Versions 7.2x

Informix GLS database servers create *GLS databases*. If you do not set the **CLIENT_LOCALE** and **DB_LOCALE** environment variables, a GLS database server creates an English database (a database with the default locale, U.S. English). For more information on these GLS environment variables, see "GLS Environment Variables" on page 10-8.

GLS Client Products

A client product (such as INFORMIX-ESQL/C) that uses the GLS feature for its language support is called a *GLS client product*. A client application uses the GLS feature to provide end-user formats for date, time, monetary, and numeric values.

Informix GLS includes the following client products:

- Version 9.x ESQL/C
- Version 8.2 ESQL/C
- Version 7.2x ESQL/C and ESQL/COBOL
- Version 7.1TD1 ESQL/C and ESQL/COBOL for Win32
- Version 5.x (WC1 and later) ESQL/C for Windows

GLS Locales

A GLS database server uses a GLS locale to provide language support for a database. A *GLS locale* is a set of Informix files that bring together the information about data that is specific to a particular culture, language, or territory. In particular, a GLS locale provides the following information:

- The name of the code set that the application data uses
- The collation order to use for character data
- The format for different types of data to appear to end users

A GLS locale file groups locale-specific information into the following locale categories.

Locale Category	Description
CTYPE	Controls the behavior of character classification and case conversion.
COLLATION	Controls the behavior of string comparisons.
NUMERIC	Controls the behavior of non-monetary numeric end-user formats.

(1 of 2)

Locale Category	Description
MONETARY	Controls the behavior of currency end-user formats.
TIME	Controls the behavior of date and time end-user formats.
MESSAGES	Controls the definitions of affirmative and negative responses to messages.

(2 of 2)

For GLS databases, the database server stores a condensed version of the database locale in the following two rows of the **systables** system catalog table:

- The row with **tabid** 90 stores the COLLATION category of the database locale.

  The COLLATION category of a locale determines the order in which the characters of the code set collate. The database server uses the COLLATION category of the database locale to collate character data. The **tabname** value for this row is GLS_COLLATE.

- The row with **tabid** 91 stores the CTYPE category of the database locale.

  The CTYPE category of a locale determines how characters of the code set are classified. The database server uses character classification for case conversion and some regular-expression evaluation. The **tabname** value for this row is GLS_CTYPE.

The rows with the values 90 and 91 in the **tabid** column of the **systables** system catalog table store the condensed locale name in the **site** column.

GLS Environment Variables

Figure 10-1 shows the locale environment variables that GLS products support.

Figure 10-1
GLS Environment Variables

GLS Environment Variable	Purpose
CC8BITLEVEL	Specifies how the C compiler handles multibyte characters.
CLIENT_LOCALE	Specifies the name of the client locale.
DB_LOCALE	Specifies the name of the database locale.
ESQLMF	Specifies whether to invoke the ESQL/C multibyte filter, **esqlmf**.
GL_DATE	Supports extended format strings for international formatting of DATE values.
GL_DATETIME	Supports extended format strings for international formatting of DATETIME values.
GLS8BITFSYS	Specifies how to handle filenames that contain non-ASCII characters.
SERVER_LOCALE	Specifies the name of the database server locale.

For backward compatibility, GLS products also support all of the NLS environment variables (see Figure 10-3 on page 10-13) and a subset of the ALS environment variables (see Figure 10-5 on page 10-16).

Performing Code-Set Conversion

A client application performs code-set conversion when the client locale does not match the database locale. In Figure 10-2, the **Code-set Conversion** column shows the environment variables that the NLS and ALS client products use to perform code-set conversion. The **Client Locale** and **Database Locale** columns show the environment variables from the client application that the GLS database server uses to set the client and database locale, respectively.

Figure 10-2
Locale Information Sent to GLS Database Servers

	GLS Database Server		
Client Product	**Client Locale**	**Database Locale**	**Code-Set Conversion**
GLS client products: (For a list of products, see "GLS Client Products" on page 10-6.)	CLIENT_LOCALE	DB_LOCALE	Between CLIENT_LOCALE and DB_LOCALE
NLS client products: 8.1x ESQL/C 6.x through 7.1x ESQL/C and ESQL/COBOL	LANG, LC_* (with DBNLS=1)	LANG, LC_* (with DBNLS=1)	Between DBAPICODE and LANG, LC_* (WITH DBNLS=1)
7.1TC1 ESQL/C for Win32 5.X (PRE-WG1) ESQL/C for Windows	LANG, LC_* (with DBNLS=1)	LANG, LC_* (with DBNLS=1)	Between CLIENT_LOCALE and DB_LOCALE
ALS client products: Version 6.x ALS ESQL/C	CLIENT_LOCALE	DB_LOCALE	Between CLIENT_LOCALE and DB_LOCALE
Version 5.x ALS ESQL/C	DBCODESET	DBCODESET	Between DBAPICODE and DBCODESET

Tip: *In Figure 10-2, LC_* is an abbreviation for the following NLS environment variables:* ***LC_CTYPE***, ***LC_COLLATE***, ***LC_MONETARY***, ***LC_NUMERIC*** *and* ***LC_TIME***. *For more information, see "NLS Environment Variables" on page 10-13.*

For more information on code-set conversion, see the *Informix Guide to GLS Functionality*.

NLS

Native Language Support (NLS)

The NLS feature provides support for single-byte, non-English data in a database application. These code points are the ASCII code set. If a code set contains more than 128 characters, some of its characters are 8-bit characters, which indicates that the eighth bit of the byte is set. Both ASCII and 8-bit characters are single-byte characters.

Tip: *NLS products do not support multibyte code sets. Previously, users of multibyte code sets relied on locally customized versions of the products such as the Informix ALS products. For more information, see "Asian Language Support (ALS)" on page 10-15.*

This section provides the following information about the NLS feature

- Which Informix products support the NLS feature?
- What is an NLS locale?
- What are the NLS environment variables?
- How do Informix GLS products support NLS products?

For more information about how to use NLS products, refer to the *Informix Guide to SQL: Reference*, Version 7.1; the *INFORMIX-ESQL/C Programmer's Manual*, Version 6.0; and the *INFORMIX-ESQL/COBOL Programmer's Manual*, Version 6.0.

Informix NLS Products

Informix NLS products are those that use the NLS feature to provide support for non-English database applications that use single-byte code sets. Informix supports the NLS feature in the following types of products:

- Database server products
- Client products

Informix introduced Native Language Support (NLS) in its Version 6.0 products.

NLS Database Servers

A database server that uses the NLS feature for its language support is called an *NLS database server*. A database server uses the NLS feature to provide the code set for data, collation order, and character classification. As Figure 10-8 shows, the following database servers are NLS database servers:

- INFORMIX-OnLine XPS, all versions up to (but not including) Version 8.2
- INFORMIX-OnLine, Version 6.0 through Version 7.1x
- INFORMIX-SE, Version 6.0 through Version 7.1x
- INFORMIX-OnLine Workgroup Server, Version 7.12

Informix NLS database servers create *NLS databases*. If you do not set the **LANG** and **DBNLS** environment variables, an NLS database server creates an English database (one that uses the default operating-system locale). For more information on these NLS environment variables, see "NLS Environment Variables" on page 10-13.

NLS Client Products

A client product (such as INFORMIX-ESQL/C) that uses the NLS feature for its language support is called an *NLS client product*. A client application uses the NLS feature to provide end-user formats for date, time, monetary, and numeric values.

Informix NLS client products include the following list:

- Version 7.1.x ESQL/C and ESQL/COBOL
- Version 6.x ESQL/C and ESQL/COBOL
- Version 7.1TC1 ESQL/C for Win32 and ESQL/COBOL for Win32
- Version 5.x (pre-WG1) ESQL/C for Windows

NLS Locales

An NLS product uses locales that are native to the operating system to provide language support. These locales are called *NLS locales* (*or operating-system locales*). An NLS locale groups locale-specific information into the following locale categories.

Locale Category	Description
CTYPE	Controls the behavior of character classification and case conversion.
COLLATION	Controls the behavior of string comparisons.
NUMERIC	Controls the behavior of non-monetary numeric end-user formats.
MONETARY	Controls the behavior of currency end-user formats.
TIME	Controls the behavior of date and time end-user formats.

For NLS databases, the database server stores the name of the database locale in the following two rows of the **systables** system catalog table:

- The row with **tabid** 90 stores the COLLATION category of the database locale.

  The COLLATION category of a locale determines the order in which the characters of the code set collate. The database server uses the COLLATION category of the database locale to collate character data. The **tabname** value for this row is NLSCOLL.

- The row with **tabid** 91 stores the CTYPE category of the database locale.

  The CTYPE category of a locale determines how characters of the code set are classified. The database server uses character classification for case conversion and some regular-expression evaluation. The **tabname** value for this row is NLSCTYPE.

The rows with the values 90 and 91 in the **tabid** column of the **systables** system catalog table store the condensed locale name in the **site** column.

NLS Environment Variables

An NLS product uses both operating-system environment variables and Informix NLS environment variables. Figure 10-3 lists Informix NLS environment variables.

Figure 10-3
NLS Environment Variables

NLS Environment Variable	Purpose
COLLCHAR	Allows client applications to use NLS collation.
DBAPICODE	Allows client applications to use a different code set than the one that the database server uses.
DBDATE	Specifies an end-user format for values in DATE columns.
DBLANG	Specifies the location of product-specific message files.
DBMONEY	Specifies an end-user format for values in MONEY columns.
DBNLS	Enables NLS features.
DBTIME	Specifies an end-user format for values in DATETIME columns.
LANG	Specifies the operating-system locale for your NLS features.
LC_COLLATE	Specifies a collating sequence for your locale-sensitive data.
LC_CTYPE	Affects the behavior of regular expressions and character-evaluation functions.
LC_MONETARY	Specifies the format and national currency symbol for monetary values.
LC_NUMERIC	Specifies the format and decimal separator for numeric values.
LC_TIME	Specifies the format for national dates and times.

NLS Support by Informix GLS Products

Informix GLS products support all NLS environment variables. Therefore, Informix GLS products can provide the following support for NLS products:

- An NLS client application can access a GLS database server.

  The GLS database server receives the NLS environment variables and uses them to determine client and database locale names (**CLIENT_LOCALE** and **DB_LOCALE**). It looks for NLS-compliant GLS locales that have the same name. The client application performs any code-set conversion between the **DBAPICODE** and **LANG** environment variables. For more information, see Figure 10-2 on page 10-9.
- A GLS client application can access an NLS database server with minimal modification.

  If the **DB_LOCALE** environment variable is set, the GLS client application performs code-set conversion between the **LC_CTYPE** (or **LANG**, if **LC_CTYPE** is not set) and **DB_LOCALE** locales. If **CLIENT_LOCALE** is also set, the GLS client application performs code-set conversion between the **CLIENT_LOCALE** and **DB_LOCALE** locales.

To use the NLS feature, you *must* set the following environment variables in the client environment:

- Set the **DBNLS** environment variable to 1 or 2 to enable the NLS feature.
- Set the **LANG** environment variable to the name of the NLS locale that the NLS database uses.

The remaining NLS environments in Figure 10-3 on page 10-13 can override some or all of the basic NLS functionality that **DBNLS** and **LANG** determine. The *Informix Guide to GLS Functionality* discusses the precedence of NLS and GLS environment variables.

If you have installed a GLS database server and want your NLS database to work with this new database server, you must convert the NLS database to a GLS database. For more information, see "Converting from NLS to GLS Language Support" on page 10-28.

Asian Language Support (ALS)

The ALS feature provides support for multibyte Asian data in a database application. If a character set contains more than 256 characters, the code set must contain multibyte characters. A *multibyte characte*r might require from 2 to 4 bytes of storage.

Tip: *ALS products support only multibyte code sets for several Asian languages. Previously, users of single-byte code sets relied on the Informix NLS products. For more information, see "Native Language Support (NLS)" on page 10-10.*

This section provides the following information about the ALS feature

- Which Informix products support the ALS feature?
- What is an ALS locale?
- What are the ALS environment variables?
- How do Informix GLS products support ALS products?

For more information about how to use ALS products, refer to your Informix ALS documentation.

Informix ALS Products

Informix ALS products use the ALS feature to provide support for non-English database applications that use Asian multibyte code sets. Before Version 7.2x, the following locally customized versions of Informix products supported Asian languages that use multibyte characters:

- Version 4.x ASCII Japanese (4.s)
- Informix ALS products, Versions 4.x, 5.x, and 6.x

An OnLine or SE database server that uses the ALS feature for its language support is called an *ALS database server*. Informix ALS database servers create *ALS databases*. A client product that uses the ALS feature for its language support is called an *ALS client product*.

ALS Locales

An ALS product includes Asian locales for Asian-language support. For ALS databases, the database server stores a condensed version of the database locale in the **systables** system catalog table. The location of this information depends on the version of the Informix ALS product, as Figure 10-4 shows.

Figure 10-4
Location in systables of ALS Locale Information

ALS Version Number	Location in systables
Version 6.x	In the rows whose **tabid** values are 95 and 96
Version 5.x	In the rows whose **tabid** values are 95, 96, and 97
Version 4.x	In a row whose **tabid** value is 98
Version 4.x ASCII	None (locale is always Japanese with the SJIS code set)

ALS Environment Variables

Figure 10-5 lists Informix ALS environment variables.

Figure 10-5
ALS Environment Variables

ALS Environment Variable	Purpose	Supported by GLS Products?
ALS8BITFSYS	Enables ALS products to handle 8-bit filenames.	No (use GLS8BITFSYS instead)
CC8BITLEVEL	Specifies the preprocessing format for the ESQL/C compiler.	Yes
CLIENT_LOCALE	For Version 5.x ALS, specifies the code set that the client application uses. For Version 6.x ALS, specifies the locale that the client application uses.	Yes

(1 of 2)

ALS Environment Variable	Purpose	Supported by GLS Products?
DBCODESET	Similar to CLIENT_LOCALE and DB_LOCALE; used by Version 5.x ALS.	No
DBCSOVERRIDE	Forces DB_LOCALE values to override default restrictions on how to access databases.	No
DBDATE	Specifies an end-user format for values in DATE columns.	Yes (use GL_DATE instead)
DBFORMAT	Specifies formats for INFORMIX-SQL and INFORMIX-4GL.	No
DBINFXRC	Specifies the pathname of a user-defined environment-variable configuration file.	No
DBLANG	Specifies the location of product-specific message files.	Yes
DB_LOCALE	For Version 5.x ALS, specifies the code set that locale-sensitive data in the database uses. For Version 6.x ALS, specifies the locale of the database. You must set the **DB_LOCALE** environment variable to store and access multibyte Asian characters in an ALS database. Otherwise, the database server assumes the locale to be U.S. English (**en_US.8859-1**).	Yes
DBMONEY	Specifies an end-user format for values in MONEY columns.	Yes
DBTIME	Specifies an end-user format for values in DATETIME columns.	Yes

(2 of 2)

ALS Support by Informix GLS Products

Informix GLS client applications support most ALS environment variables. (For more information, see Figure 10-5 on page 10-16.) Therefore, Informix GLS products provide the following support for ALS products:

- An ALS client application can access a GLS database server.

  For information on how a GLS database server supports ALS client applications, see Figure 10-6.
- A GLS client application can access an ALS database server.

  For information on how GLS client applications support connections to ALS database servers, see Figure 10-7 on page 10-19.

Figure 10-6
GLS Database Server Support for ALS Client Applications

ALS Client Application	Support in GLS Database Server
Version 6.x ALS ESQL/C applications	The database server obtains the database locale name from the **DB_LOCALE** environment variable. The client application performs code-set conversion between the locales that **CLIENT_LOCALE** and **DB_LOCALE** specify.
Version 5.x ALS ESQL/C applications	The database server obtains the database locale name from the **DBCODESET** environment variable. It converts this value to a valid GLS locale name and uses this locale name for **DB_LOCALE**. The client application performs code-set conversion between the locales that **DBAPICODE** and **DBCODESET** specify.

(1 of 2)

ALS Client Application	Support in GLS Database Server
Version 4.x ALS ESQL/C applications, Version 4.x ALS Viewpoint	The database server obtains the database locale name from the **DBCODESET** environment variable. It converts this value to a valid GLS locale name and uses this locale name for **DB_LOCALE**. The client application cannot perform code-set conversion.
Version 4.x ASCII ESQL/C applications	The database server obtains the database locale name from the **DB_LOCALE** environment variable *on the server computer*. This environment variable must be set to the Japanese locale, **ja_jp.sjis**, *before* the database server is initialized. The client application does not need to perform code-set conversion.

(2 of 2)

Figure 10-7
GLS Client Applications Support for ALS Database Servers

ALS Database Server	Support in GLS Client Application
Version 6.x ALS database server	You must set the **DB_LOCALE** environment variable to the name of the database locale for the ALS database.
Version 5.x ALS database server	You must set the **CLIENT_LOCALE** and **DB_LOCALE** environment variables if the GLS client application is to perform code-set conversion. The **DBCODESET** environment variable enables the connection. This environment does not affect the client application but is sent on to the ALS database server for processing.
Version 4.x ALS database server	You must set the **DBCODESET** environment variable to the database locale name for the ALS database server. The client application cannot perform code-set conversion.
Version 4.x ASCII database server	You must set both **CLIENT_LOCALE** and **DB_LOCALE** environment variables to the Japanese locale, **ja_jp.sjis**.

The remaining ALS environments in Figure 10-5 on page 10-16 can override some or all of the basic ALS functionality. The *Informix Guide to GLS Functionality* discusses the precedence of ALS and GLS environment variables.

If you have installed a GLS database server and want your ALS database to work with this new database server, you must ensure that the GLS database server can support the ALS database. For more information, see "Converting from ALS to GLS Language Support" on page 10-35.

Language Support in Informix Database Servers

When a database server creates a database, it assigns a *database locale* to that database. The database locale determines the following information for the database:

- The code set whose characters are valid in any character column
- The code set whose characters are valid in the names of database objects such as databases, tables, columns, and views
- The localized order to collate data from any NCHAR and NVARCHAR columns (NLS products only)

Informix products use the database locale when they create new databases or read existing databases. A database inherits its locale from the session that creates it. When an application development tool issues the CREATE DATABASE statement, the database server creates a new database and assigns it a database locale.

The database server saves the database locale name in the system catalog of the database. Special rows in the **systables** system catalog table contain information that is used throughout the lifetime of the database for such operations as handling regular expressions, collating character strings, and ensuring proper use of code sets. At runtime, the database server maps this database locale to a locale file on the system where the database resides.

When you migrate an Informix database server whose databases contain non-English data, you need to consider whether the upgrade involves a change in Informix language support. Such a change might require:

- a migration of the database data.
- a change in the locale that the database uses.
- a change in the way locale information is stored in the **systables** system catalog table.

If the locale of the data is not compatible with the language support that the database server uses, you might experience compatibility problems. Figure 10-8 shows the language support that different Informix database servers use.

Figure 10-8
Language Support In Informix Database Servers

Database Server Version	Language Support
All versions	English
Universal Server and Informix Dynamic Server with Universal Data Option	GLS
Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options Version 8.2 and later	GLS
INFORMIX-OnLine XPS up to Version 8.1x	English only
OnLine and SE	
Versions 7.2x	GLS
Versions 6.x through 7.1x	NLS
Pre-Version 6.0	English only
Dynamic Server, Workgroup and Developer Editions, Version 7.3	GLS
OnLine Workgroup Server Versions 7.2 Version 7.12	NLS
OnLine and SE (with ALS) Versions 4.x, 5.x, 6.x	ALS

The following sections provide a brief summary of the support that Informix products provide for languages other than English.

Universal Server and Universal Data Option

INFORMIX-Universal Server and Informix Dynamic Server with Universal Data Option uses the GLS feature to support languages that use both single-byte and multibyte characters. When you upgrade Universal Data Option, you do *not* change language support and therefore do not need to migrate databases.

INFORMIX-OnLine XPS and Advanced Decision Support and Extended Parallel Options

The language support that INFORMIX-OnLine XPS and Dynamic Server with AD and XP Options use depends on the version of the database server, as follows:

- Version 8.2 and later products use the GLS feature to support languages that use both single-byte and multibyte characters.
- Versions through Version 8.1x use English only.

The following table summarizes whether you need to migrate databases when you upgrade your OnLine XPS or Dynamic Server with AD and XP Options database server.

Migrate from	Migrate to	Change language support?	For more information
Version 8.2x and later	Version 8.2x and later	No	You do not need to migrate databases.
Versions through Version 8.1x	Versions through Version 8.1x	No	You do not need to migrate databases.
Versions through Version 8.1x	Version 8.2x and later	From English only to GLS	You do not need to migrate databases.

Informix Dynamic Server and INFORMIX-OnLine Dynamic Server

The language support that Informix Dynamic Server and INFORMIX-OnLine Dynamic Server use depends on the version of the database server, as follows:

- Version 7.2x and later use the GLS feature to support languages that use both single-byte and multibyte characters.
- Version 6.0 through Version 7.1x use the NLS feature to support single-byte native languages.
- Before Version 6.0, support for languages other than English required locally customized versions of Informix products.

  Versions 4.x, 5.x, and 6.x of Informix ALS products provided support for multibyte characters. The standard English versions of pre-6.0 Informix products do not accommodate alternative languages. These products use the default locale of the operating system for their language support. In most cases, the default locale of the operating system is English.

The following table summarizes whether you need to migrate databases when you upgrade your Informix Dynamic Server or OnLine database server.

Migrate from	Migrate to	Change language support?	For more information
Version 7.2x and later	Version 7.2x and later	No	You do not need to migrate databases.
Version 6.0 through Version 7.1x	Version 6.0 through Version 7.1x	No	You do not need to migrate databases.
Version 6.0 through Version 7.1x	Version 7.2x and later	From NLS to GLS	"Converting from NLS to GLS Language Support" on page 10-28

(1 of 2)

Migrate from	Migrate to	Change language support?	For more information
Before Version 6.0 (Informix ALS products only)	Version 7.2x and later	From ALS to GLS	"Converting from ALS to GLS Language Support" on page 10-35.
Before Version 6.0 (English-language)	Version 7.2x and later	No (English to English)	You do not need to migrate databases.

(2 of 2)

INFORMIX-SE

Versions of the INFORMIX-SE database server use the same language support as the corresponding OnLine Dynamic Server version. For more information, see "Informix Dynamic Server and INFORMIX-OnLine Dynamic Server" on page 10-23.

Informix Dynamic Server, Workgroup and Developer Editions and INFORMIX-OnLine Workgroup Server

The language support that Dynamic Server, Workgroup and Developer Editions and INFORMIX-OnLine Workgroup Server uses depends on the version of the database server, as follows:

- Version 7.2 and later of Dynamic Server, Workgroup and Developer Editions or OnLine Workgroup Server products use the GLS feature to support languages that use both single-byte and multibyte characters.
- Versions through Version 7.12x of OnLine Workgroup Server use the NLS feature to support single-byte native languages.

The following table summarizes whether you need to migrate databases when you upgrade your Dynamic Server, Workgroup and Developer Editions or OnLine Workgroup Server database server.

Migrate from	Migrate to	Change language support?	For more information
Version 7.2 and later	Version 7.2 and later	No	You do not need to migrate databases.
Versions through Version 7.12	Versions through Version 7.12	No	You do not need to migrate databases.
Versions through Version 7.12	Version 7.2 and later	From NLS to GLS	"Converting from NLS to GLS Language Support" on page 10-28

Converting to GLS Language Support

If you have an ALS or NLS database that you want to convert to run under a GLS database server, perform the following steps:

1. Verify that the GLS database server supports the database locale of the database you want to convert.
2. Convert the NLS or ALS database to a GLS database.

Determining Existing Locale Support

To verify that a GLS database server supports the database locale of your NLS or ALS database, you need to know:

- the current database locale that your database uses.
- what locales are available to your GLS database server.

Finding the Current Database Locale

The database locale information is stored in the system catalog of the NLS or ALS database, in special rows of the **systables** system catalog table. The exact location of this information depends on the particular database server you currently use, as follows:

- The NLS database stores database locale information in **systables** rows with **tabid** values of `90` and `91`.
- The **systables** rows that an ALS database uses to store database locale depends on the version of the ALS database server. For more information, see Figure 10-4 on page 10-16.

You can use a SELECT statement to query the **systables** system catalog table for the database locale information of a database.

UNIX

Finding Available GLS Locales on UNIX

To find out what GLS locales are available on your UNIX system, you can run the **glfiles** utility. This utility checks the GLS system directories and creates a file listing for the locale files that these directories contain. The file listing is in a file called **lc*X*.txt**, where *X* is the version of the locale files. For more information on **glfiles**, refer to the *Informix Guide to GLS Functionality*.

To run the glsfiles utility

1. Set your **INFORMIXDIR** and **PATH** environment variables to point to a GLS database server.
2. Execute the **glfiles** utility.
3. Examine the **lc*X*.txt** file for a GLS locale name that is compatible with your NLS locale. If you do not find an operating-system-compatible locale file for your NLS locale, try to find the closest fit.

Important: *The **glfiles** utility is only available on UNIX platforms with products that support the GLS feature.*

WIN NT

Finding Available GLS Locales on Windows NT

To find out which GLS locales are available on your Windows system, you must look in the GLS system directories. A GLS locale resides in the following file:

```
%INFORMIXDIR%\gls\lcX\lg_tr\codemodf.lco
```

In this path, **INFORMIXDIR** is the environment variable that specifies the directory in which you install the Informix product, **gls** is the subdirectory that contains the GLS system files, *X* represents the version number for the locale file format, *lg* is the 2-character language name, *tr* is the 2-character territory name that the locale supports, and *codemodf* is the condensed locale name. For more information on the location of the GLS locale files, see the *Informix Guide to GLS Functionality*.

Verifying GLS Database Locales

To find the database locales for all databases that your GLS database server supports, execute the following command in a GLS DB-Access:

```
SELECT * FROM sysmaster:sysdbslocale
```

The **sysdbslocale** system catalog table has the following two columns:

- The **dbs_dbsname** column holds the name of the database.
- The **dbs_collate** column holds the name of the database locale for the database that the **dbs_dbsname** column specifies.

For example, suppose that your GLS database server (on a UNIX system) supports the following databases: **stores7** and **accounting**. If the **stores7** database uses the default locale (**en_us.8859-1**) and the **accounting** database uses the French locale (**fr_fr.8859-1**), the preceding query might return the following output:

```
dbs_dbsname dbs_collate

sysmaster   en_US.819
sysutils    en_US.819
stores7     en_US.819
accounting  fr_FR.819
```

Converting from NLS to GLS Language Support

NLS databases use locales that the operating system provides and single-byte code sets. GLS databases use locales that Informix provides. Before you use an NLS database with a GLS database server, follow these steps:

1. Determine the database locale name of the current NLS database.
2. Determine which GLS locale is compatible with the current NLS database locale.
3. Open the NLS database with the GLS database server.

Important: *Check the availability of the GLS locale before you open NLS databases with a GLS database server. A GLS database server cannot open an NLS database that has an unsupported locale.*

This section discusses how to change from NLS language support to GLS language support when you migrate from an NLS database server to a GLS database server.

Warning: *Do not overwrite your NLS database server with the GLS database server if you want to migrate a non-English NLS database to a GLS database. Migration of a non-English NLS database requires that both the NLS and GLS database servers be installed.*

Determining the Locale of the NLS Database

To determine the current database locale for an NLS database, examine the site column for rows 90 and 91 of the **systables** system catalog. For more information on the use of **systables** by an NLS database, see "NLS Locales" on page 10-12.

To determine the database locale of the NLS database

1. Set the appropriate NLS environment variables (see Figure 10-3 on page 10-13).
2. Start an NLS database server.
3. Launch DB-Access.
4. Open your NLS database.

5. Use the DB-Access Query option to execute one of the following commands, depending on your database server type.

ODS

OWS

XPS

To see the locale names of the COLLATION and CTYPE categories of a database locale for an OnLine XPS, OnLine Dynamic Server, or OnLine Workgroup Server database, use the following SELECT statement:

```
SELECT tabname, site FROM systables
WHERE tabid = 90 OR tabid = 91
```

The value displayed in the **site** column is the name of your current NLS locale.

The following SELECT statement returns the name of the current database locale for each database that the NLS database server supports:

```
SELECT UNIQUE dbsname, collate
FROM sysmaster:systabnames
```

The **dbsname** column lists the databases on your database server. The entry in the **collate** column shows the NLS locale of the database. If the **collate** column is blank, the NLS database is an English database. ♦

SE

To see the locale names of the COLLATION and CTYPE categories for a database locale of an SE database, use the following SELECT statement:

```
SELECT dirpath FROM systables
WHERE tabid = 90 OR tabid = 91
```

The value in the **dirpath** column is the name of your current NLS locale. ♦

If you do *not* find a matching entry in **systables**, your database uses the default locale for the operating-system, which is usually U.S. English. An English NLS database does *not* require conversion. For more information, see "To open an English NLS database" on page 10-34.

Determining Whether a Compatible GLS Locale Exists

To determine whether a compatible GLS locale exists for your non-English NLS database, you need to determine the following information:

- Which type of GLS locale do you want to use?
- Which GLS locales are installed on your system?

GLS products provide the following two types of locales:

- **Operating-system-independent locales**. The operating-system-independent locales are the locales that Informix has developed to provide a more portable way to support culture-specific information. With these locales, GLS products can access culture-specific information regardless of the operating system under which they run.
- **Operating-system locales**. The operating-system locales are special GLS locales that are compatible with the locales native to different operating systems. For a given language and territory, operating-system locales might have different definitions from one operating system to another. GLS products provide operating-system locales for backward compatibility with NLS databases.

Tip: *GLS operating-system-independent locales are usually referred to simply as "GLS locales." The operating-system locales are also GLS locales; they share the same format as the operating-system-independent locales. However, they are useful only for NLS databases that want to continue use of their NLS locale. In this manual, the term "GLS locale" refers to an operating-system-independent locale.*

To migrate a non-English NLS database to a GLS database, you need to decide which of these types of locales to use. Your decision is not irreversible. You can also change the locale after you migrate to a GLS database server.

To use an operating-system locale

An upgrade of your NLS database to a GLS database that uses an operating-system locale requires little special action on your part.

Important: *Use of an operating-system locale can result in incorrect results when you perform distributed queries across dissimilar environments. The locale-category definitions of one operating-system locale might differ from the locale-category definitions of the same locale on another operating system or in a GLS locale.*

To use the operating-system locale that is compatible with your current NLS database locale, perform the following steps:

1. Confirm that an operating-system locale exists for your current NLS database locale.

   For more information, see "Finding Available GLS Locales on UNIX" on page 10-26.

2. Set the appropriate NLS environment variables, such as **LANG** and **DBNLS**, to indicate the NLS locale.

   For more information on NLS environment variables, see "NLS Environment Variables" on page 10-13.

3. Open the NLS database with a GLS database server.

   For more information, see "To open a non-English NLS database" on page 10-33.

You do not need to perform any additional steps on the data or the system catalog. If you set the **CLIENT_LOCALE** or **DB_LOCALE** environment variables, make sure that you use the operating-system locale as the locale name.

Warning: *If your NLS database has a database locale name that is not a valid operating-system locale, or if a GLS environment variable (**CLIENT_LOCALE** or **DB_LOCALE**) is not a valid operating-system locale, the GLS database server cannot open it. You must set the **LANG** and **DBNLS** environment variables correctly for the GLS database server to open a non-English NLS database.*

To use a GLS (operating-system-independent) locale

A conversion of your non-English NLS database to a GLS database that uses a GLS (operating-system-independent) locale requires modification of the data and the system catalog. To use a GLS locale for your database, perform the following steps:

1. Determine which GLS locale is compatible with your current database locale.

   For more information, see "Finding Available GLS Locales on UNIX" on page 10-26.

2. Set the appropriate NLS environment variables, such as **LANG** and **DBNLS**, to indicate the NLS locale.

   Do *not* set the GLS environment variables, **CLIENT_LOCALE** and **DB_LOCALE**, to the name of the new GLS locale. If you set one of these GLS environment variables, use the name of the appropriate operating-system locale. For more information on NLS environment variables, see "NLS Environment Variables" on page 10-13.

3. Unload the NLS database with the **dbexport** utility.

   When you unload data from an NLS database that uses an English-language locale, the resulting text files are ASCII files. When you unload data from an NLS database that uses a non-English-language locale, the resulting text files might include 8-bit characters and multibyte characters. For more information on how to use **dbexport**, see "Using the dbexport and dbimport Utilities" on page 9-6.

4. Drop the NLS database.

   You must drop the old NLS database before you import the data so that the **dbimport** utility can create a GLS database with the same name.

5. Set the GLS environment variables, **CLIENT_LOCALE** and **DB_LOCALE**, to the name of the GLS database locale that you want to use.

   For more information on GLS environment variables, see "GLS Environment Variables" on page 10-8.

6. Create a new GLS database and load the data into this database with the **dbimport** utility.

   The **dbimport** utility determines the database locale from the **DB_LOCALE** environment variable and makes the appropriate corrections to the system catalog tables. For more information on how to use **dbimport**, see "Using the dbexport and dbimport Utilities" on page 9-6.

You can now open the new GLS database with a GLS database server. You no longer need to set the NLS environment variables.

Opening the NLS Database

If you choose to use an operating-system locale for your NLS database, a GLS database server expects the NLS environment variables to indicate the locale of the database, as follows:

- For an English NLS database, no NLS environment variables need to be set.

  The default locale on most operating systems is U.S. English.
- For a non-English NLS database, the **LANG** and **DBNLS** environment variables must be set to indicate the operating-system locale

  You can set other NLS environment variables to provide additional locale information.

With the NLS environment variables correctly set, you can open an NLS database with a GLS database server. The GLS database server automatically updates the **systables** system catalog table with its own database locale information when it opens the NLS database. For more information about the GLS database locale information, see "GLS Locales" on page 10-6.

To open a non-English NLS database

If you choose to use an operating-system locale for your non-English NLS database, you must *always* set the NLS environment variables (such as **LANG** and **DBNLS**) appropriately. If you do *not* set these NLS environment variables, neither a GLS database server nor GLS database utilities can access the database. If you also set a GLS environment variable (**CLIENT_LOCALE** or **DB_LOCALE**), you must ensure that it is set to the name of the appropriate operating-system locale.

To open a non-English NLS database, follow these steps:

1. Set the NLS environment variables, such as **LANG** and **DBNLS**, to indicate use of an operating-system locale for the database.

   For more information on NLS environment variables, see Figure 10-3 on page 10-13.
2. You can optionally set the GLS environment variables, **CLIENT_LOCALE** and **DB_LOCALE**, to the name of the operating-system locale for the database.

   For more information on GLS environment variables, see Figure 10-1 on page 10-8.

3. When the GLS database server opens an NLS database, it automatically updates the **systables** system catalog table, as follows:

   - In the row whose **tabid** value is 90, the database server changes the **tabname** value from NLSCOLL to GL_COLLATE.
   - In the row whose **tabid** value is 91, the database server changes the **tabname** value from NLSCTYPE to GL_CTYPE.
   - The database server creates a condensed database locale name from the locale name in rows 90 and 91 of **systables**.

   If a locale exists that corresponds to the condensed name, the database server stores the condensed locale name in rows 90 and 91. If the existing locale does *not* have a condensed locale name, the database server does not change the locale name in rows 90 and 91.

4. If the GLS locale defines a localized collation order, the GLS database server converts the data type of all CHAR and VARCHAR columns in the system catalog tables to NCHAR and NVARCHAR.

   If the GLS locale does not define a localized collation order, the database server does not convert the data type of all CHAR and VARCHAR columns in the system catalog tables.

Warning: *If the* ***systables*** *rows with* ***tabid*** *values of* 90 *and* 91 *contain an invalid operating-system locale name, the GLS database server cannot open the NLS database.*

To open an English NLS database

GLS products use the default locale, U.S. English, automatically. Most NLS products also use U.S. English by default because the default locale on most operating systems is U.S. English. Therefore, you are not required to set the GLS environment variables, **CLIENT_LOCALE** and **DB_LOCALE**, before you open an English NLS database with a GLS database server. (If you do set these environment variables, they must be set to the name of the default GLS locale.)

When you open an English NLS database, the GLS database server changes the **systables** system catalog table, as follows:

- Creates a row with the **tabname** of GL_COLLATE and the **tabid** of 90.
- Creates a row with the **tabname** of GL_CTYPE and the **tabid** of 91.
- Stores a condensed locale name for the default locale in the **site** column of rows 90 and 91.

The default locale, U.S. English, does not define a localized collation order. Therefore, the GLS database server does *not* convert CHAR and VARCHAR columns in the system catalog tables to NCHAR and NVARCHAR. For English databases, the database server uses code-set collation order for collation of data in *all* character columns (in the system catalog tables or user-defined tables).

Converting from ALS to GLS Language Support

The Informix Asian Language Support (ALS) products include ALS locales, which contain culture-specific information and the multibyte code sets that Asian languages require. When a GLS database server opens an ALS database, it automatically converts the database locale name from its ALS locale name to a GLS locale name and then copies this GLS locale name into rows `90` and `91` of the **systables** system catalog table. Therefore, you do not need to change the database locale name from its ALS locale to a GLS locale when you migrate an ALS database to a GLS database.

This section discusses how to convert from ALS language support to GLS language support for the following versions of ALS databases:

- Version 6.x ALS databases
- Version 5.x ALS databases
- Version 4.x ALS databases
- Version 4.x ASCII Japanese databases

Migrating from Version 6.x ALS Products

The GLS database server automatically migrates a Version 6.x ALS database when it opens the database. Informix Version 6.x ALS databases store locale information in the **systables** system catalog table in the rows whose **tabid** values are `95` and `96`. When you use a GLS database server to open a Version 6.x ALS database, the database server copies this locale information to rows in **systables** whose **tabid** values are `90` and `91`, respectively.

To open a Version 6.x ALS database

1. Verify that the current ALS locale of your database is available to your GLS database server.

   Figure 10-9 shows the Version 6.x ALS locales. For more information on how to determine which GLS locales are installed with your database server, see "Finding Available GLS Locales on UNIX" on page 10-26. If a compatible GLS locale is *not* available, you must obtain this GLS locale *before* you proceed with the migration.

2. Set the GLS environment variables, **CLIENT_LOCALE** and **DB_LOCALE**, to the new GLS locale for the ALS database.

   For more information on GLS environment variables, see Figure 10-1 on page 10-8.

3. Set the **GLS8BITFSYS** environment variable to the same value as the **ALS8BITFSYS** environment variable.

4. When the GLS database server opens the Version 6.x ALS database, it automatically updates the **systables** system catalog table, as follows:

   - In the row whose **tabid** value is 90, the database server copies the data from the row whose **tabid** is 95 and assigns the **tabname** value of GL_COLLATE to row 90.
   - In the row whose **tabid** value is 91, the database server copies the data from the row whose tabid is 96 and assigns the **tabname** value of GL_CTYPE to row 91.

- The database server creates a condensed database locale name from the locale name in rows 90 and 91 of **systables**.

If a locale exists that corresponds to the condensed name, the database server stores the condensed locale name in rows 90 and 91. If the existing locale does *not* have a condensed locale name, the database server does not change the locale name in rows 90 and 91.

Figure 10-9
Version 6.x ALS Locale-Names

Asian Language	6.x ALS Locale Name
English	en_us.8859-1
China	zh_CN.gb
Korea	ko_KR.ksc
Japan	ja_JP.sjis ja_JP.sjis-s ja_JP.ujis
Taiwan	zh_TW.big5 zh_TW.sbig5 zh_TW.ccdc zh_TW.cccii

Migrating from Version 5.x ALS Products

The GLS database server automatically migrates a Version 5.x ALS database when it opens the database. Version 5.x ALS databases store code-set aliases in the database.

Warning: *If your 5.x ALS databases uses a* ***DBCODESET*** *of* ***sjis*** *and has user-defined double-byte characters that are mapped between* `0xf040` *and* `0xfcfc`*, the following restrictions apply:*

- *Do not open the 5.x ALS database with a GLS database server.*

  Instead, use the ***dbexport*** *and* ***dbimport*** *utilities or the UNLOAD and LOAD commands in DB-*Access *to migrate the 5.x ALS database to a GLS database.*
- *Do not issue the SQL statement CREATE DATABASE from 5.x ALS client applications that use a* ***DBCODESET*** *of* ***sjis****.*

  Instead, use a GLS version of DB-Access to create new databases. However, you can use 5.x ALS client applications to create new tables, indexes, views, and so forth.

To open a Version 5.x ALS database

1. Verify that the current ALS code-set alias of your database is available to your GLS database server.

   Figure 10-10 shows the GLS locale names for the Version 5.x code-set aliases. For more information on how to determine which GLS locales are installed for your database server, see "Finding Available GLS Locales on UNIX" on page 10-26. If the compatible GLS locale is *not* available, you must obtain this GLS locale *before* you proceed with the migration.
2. Set the GLS environment variables **CLIENT_LOCALE** and **DB_LOCALE** to the new GLS locale for the ALS database.

   For more information on GLS environment variables, see Figure 10-1 on page 10-8.

3. When the GLS database server opens the Version 5.x ALS database, it automatically performs the following steps:

- Maps the 5.x code-set alias to its GLS locale name, as Figure 10-10 shows.

  If the code-set alias stored in the 5.x ALS database is *not* one of these aliases, the conversion fails. The conversion also fails if the code-set alias was customized.

- Stores the mapped GLS locale name to rows in **systables** whose **tabid** values are 90 and 91.

  For more information on how a GLS database stores information in **systables**, see "GLS Locales" on page 10-6.

Figure 10-10
Locale-Name Conversion for a Version 5.x ALS Database

Version 5.x ALS Code-Set Name	GLS Locale Name
big5	zh_TW.big5
sbig5	zh_TW.sbig5
ccdc	zh_TW.ccdc
cccii	zh_TW.cccii
8859-1	en_us.8859-1
gb2312	zh_CN.gb
ks5601	ko_KR.ksc
jis208	ja_jp.ujis
sjis208	ja_jp.sjis

Migrating from Version 4.x ALS Products

The GLS database server automatically migrates a Version 4.x ALS database when it opens the database. Informix Version 4.x ALS databases store locale information in the **systables** system catalog table in a row whose **tabid** value is `98`. When you use a GLS database server to open a Version 4.x ALS database, the database server copies this locale information to rows in **systables** whose **tabid** values are `90` and `91`.

Warning: *If your 4.x ALS database uses a* ***DBCODESET*** *of* ***sjis*** *and has user-defined double-byte characters that are mapped between* `0xf040` *and* `0xfcfc`*, the following restrictions apply:*

- *Do not open this 4.x ALS database with a GLS database server.*

  Instead, use the ***dbexport*** *and* ***dbimport*** *utilities or UNLOAD and LOAD commands in DB-*Access *to migrate the 4.x ALS database to a GLS database.*

- *Do not issue the SQL statement CREATE DATABASE from 4.x ALS client applications that use a* ***DBCODESET*** *of* ***sjis****.*

  *Instead, use a GLS version of DB-*Access*, to create new databases. However, you can use 4.x ALS client applications to create new tables, indexes, views, and so forth.*

To open a Version 4.x ALS database

1. Verify that the current ALS code-set name of your database is available to your GLS database server.

   Figure 10-11 shows the GLS locale names for the Version 4.x code-set names. For more information on how to determine which GLS locales are installed for your database server, see "Finding Available GLS Locales on UNIX" on page 10-26. If the compatible GLS locale is *not* available, you must obtain this GLS locale *before* you proceed with the migration.

2. Set the GLS environment variables **CLIENT_LOCALE** and **DB_LOCALE** to the new GLS locale for the ALS database.

   For more information on GLS environment variables, see Figure 10-1 on page 10-8.

3. When the GLS database server opens the Version 4.x ALS database, it automatically performs the following steps:

   - Maps the 4.x code-set name that is stored in row 98 of the **systables** system catalog table to a GLS locale name, as Figure 10-11 shows.

     If the code set that is stored in the 4.x ALS database is *not* one of these names, the conversion fails.

   - Stores the mapped GLS locale name in the **site** column of the **systables** rows whose **tabid** values are 90 and 91.

     For more information on how a GLS database stores information in **systables**, see "GLS Locales" on page 10-6.

Figure 10-11
Locale-Name Conversion for a Version 4.x ALS Database

4.1 ALS Code-Set Name	GLS Locale Name
ascii	en_us.8859-1
gb	zh_CN.gb
ksc	ko_KR.ksc
ksc1	ko_KR.ksc
big5	zh_TW.big5
sbig5	zh_TW.sbig5
ccdc	zh_TW.ccdc
cccii	zh_TW.cccii

Migrating from Version 4.x ASCII Japanese Products

The GLS database server cannot automatically convert 4.x ASCII databases (Japanese-language version of 4.x ALS) to GLS databases. You must use the **dbexport** and **dbimport** utilities or the UNLOAD and LOAD commands to unload the 4.x ASCII database and then load it into a GLS database server.

Warning: *Do not issue the SQL statement CREATE DATABASE from 4.x ASCII client applications. Instead, use a GLS version of DB-Access to create new databases. However, you can use 4.x ASCII client applications to create new tables, indexes, views, and so forth.*

Reverting from GLS Language Support

The **onmode** utility with its **-b** option reverts a GLS database with the following syntax:

```
onmode -b reversion_level
```

In the preceding syntax, *reversion_level* indicates to which version of the database server (and hence to which language support) **onmode** reverts all databases that the GLS database server supports. For more information on the **onmode** utility, see your GLS database server *Administrator's Guide*.

The following sections describe how to use **onmode** to revert from a GLS database to:

- an NLS database.
- an ALS database.

Reverting from GLS to NLS Language Support

During the reversion process, the **onmode** utility moves the database locale information in **systables** from its GLS location to its NLS location.

Warning: *Before you revert a GLS database to an NLS database, verify that the GLS databases use operating-system locales. Also verify that your operating system supports these locales. An NLS database server can open a reverted GLS database only if the operating system supports the database locale.*

This section discusses how to revert a GLS database to an NLS database for use with the following NLS database servers:

- OnLine Dynamic Server and OnLine Workgroup Server
- SE database server

Reverting to an OnLine NLS Database Server

NLS database servers include OnLine Dynamic Server 6.0, 7.1, 7.1UD1, and 7.10.UCx, and OnLine Workgroup Server 7.12 support only the NLS feature for language support.

To revert from a GLS database to an NLS database

1. Use the **onmode** utility with its **-b** option, as follows:

   ```
   onmode -b reversion_level
   ```

   In the previous syntax, *reversion_level* is one of the values shown in the following table.

OnLine Target Version	*reversion_level* Value
6.0	6.0
7.1UC1	7.1
7.1UD1, 7.11, 7.12, 7.13, or 7.14	7.1UD1

2. Set the appropriate NLS environment variables.

   An NLS databases server ignores the **CLIENT_LOCALE**, **DB_LOCALE**, and **SERVER_LOCALE** environment variables. For more information on NLS environment variables, see Figure 10-3 on page 10-13.

If the GLS database uses the default locale (U.S. English) and none of its user-defined tables have NCHAR or NVARCHAR columns, the **onmode** utility performs the following actions to revert the database:

- It deletes the locale entries in **systables**.
- It converts any NCHAR and NVARCHAR columns in the system catalog tables to CHAR and VARCHAR columns, respectively.
- It remakes the indexes on the system catalog tables.

If the GLS database is a non-English database (it uses any nondefault locale), or if any user-defined table has NCHAR or NVARCHAR columns, the **onmode** utility changes the locale entries in **systables** to the NLS format (**tabname** from rows 90 and 91 changes from GL_COLLATE and GL_CTYPE to NLSCOLL and NLSCTYPE, respectively). You do not need to remake the indexes.

Reverting to an NLS SE Database Server

INFORMIX-SE does not provide a direct path for reverting from a GLS version of the product to earlier NLS versions. For more information, refer to Chapter 8, “Migrating SE.”

Reverting from GLS to ALS Language Support

During the reversion process, the **onmode** utility moves the database locale information in the **systables** system catalog table from its GLS location to its ALS location.

Warning: *Before you revert a GLS database to an ALS database, verify that the GLS database uses a locale that the ALS database server supports. An ALS database server can open a reverted GLS database only if the database locale is one that the database server supports.*

This section discusses how to revert a GLS database to an ALS database for use with the following Informix ALS products:

- Version 6.x ALS database servers
- Version 5.x ALS database servers
- Version 4.x ALS database servers

Reverting to Version 6.x ALS

Use the **-b 6.0A** option of the **onmode** utility to revert a GLS database to a Version 6.x ALS database. The **onmode** utility performs the following actions to revert the database:

- It moves the locale entries in **systables** to the 6.0 ALS database location (from rows 90 and 91 to rows 95 and 96).
- It converts NCHAR columns in the system catalog table to CHAR columns.
- It remakes the indexes in the system catalog tables.

***Important**: Only NCHAR and NVCHAR columns are affected by a collation order. A collation order, specified in a locale, is ignored if a the table does not contain NCHAR or NVCHAR.*

To revert from a GLS database to a Version 6.x ALS database

1. Verify that the Version 6.x ALS database server supports the current GLS database locale.

   Figure 10-9 on page 10-37 shows the Version 6.x ALS locales. If the GLS locale is *not* one of these locales, the reversion fails.

2. Choose one of the following methods to handle any NCHAR or NVARCHAR columns from your database:

   - Replace NCHAR columns with CHAR columns and NVARCHAR columns with VARCHAR columns.

     This solution means that you lose the locale-specific data, but it ensures that the 6.x ALS database server supports the remaining data.

   - Drop the NCHAR or NVARCHAR columns from your database.

     This solution means that the database server no longer collates character data in a localized order.

3. Use the following **onmode** command to perform the reversion:

   ```
   onmode -b 6.0A
   ```

4. Set the appropriate ALS environment variables.

   For more information, see Figure 10-5 on page 10-16.

Warning: *If the database has NCHAR or NVARCHAR in any user-defined table,* ***onmode*** *fails.*

Suppose you convert a 6.x ALS database to a GLS database, then revert it back to 6.x ALS. This type of reversion does not pose a problem because the database server did not remove the original ALS locale entries in rows 95 and 96 when it converted the ALS database to a GLS database.

However, when you revert a database that a GLS database server creates, the **onmode** utility might encounter some problems. A GLS database server and a Version 6.x ALS database server can support the same Asian locales. The **onmode** utility moves the locale information from rows 90 and 91 to rows 95 and 96 of **systables**. However, the GLS database stores its database locale name in a condensed format, which a Version 6.x ALS database server does not recognize.

For example, both Version 6.x ALS products and GLS products support the default locale, U.S. English. However, Version 6.x ALS products store the name of this locale as **en_us.8859-1**, while GLS products (on a UNIX platform) store it as **en_us.819**, which 6.x ALS products do not understand. You might need to update the name of the database locale in **systables** to a form that the Version 6.x ALS database server supports.

Reverting to Version 5.x ALS

Use the **-b 5.0** option of the **onmode** utility to revert a GLS database to a Version 5.x ALS database.

Important: *You cannot revert from a GLS database server to a 5.x ALS database server automatically.*

To revert from a GLS database to a 5.x ALS database

1. Verify that the Version 5.x ALS database server supports the current GLS database locale.

   Figure 10-10 on page 10-39 shows the valid GLS locales for a Version 5.x ALS database server. If the GLS locale is *not* one of these locales, the reversion fails.

1. Verify that the current GLS locale is supported in Version 5.x ALS. (See Figure 10-10 on page 10-39.)
2. Choose one of the following methods to handle any NCHAR or NVARCHAR columns from your database:
   - Drop all NCHAR and NVARCHAR columns from the database.

     This solution means that you lose the locale-specific data, but it ensures that the 5.x ALS database server supports the remaining data.

   - Convert the NCHAR columns to CHAR and NVARCHAR columns to VARCHAR.

     This solution means that the database server no longer collates character data in a localized order.

3. Use the following **onmode** command to revert the GLS databases:

   ```
   onmode -b 5.0
   ```

4. Set the appropriate ALS environment variables.

   For more information, see Figure 10-5 on page 10-16.

All databases on OnLine Dynamic Server 5.0 have to be in **en_us.8859-1** or **en_us.819** for both CTYPE and COLLATE.

If your GLS database contains locale-specific data (NCHAR and NVARCHAR columns), and you revert to a database server that does not support non-English data, you lose access to this data.

Warning: *The **onmode** utility fails when it attempts to revert a GLS database that contains NCHAR or NVARCHAR columns to a Version 5.x ALS database.*

Reverting to Version 4.x ALS or ASCII Product

GLS products do not support reversion to Version 4.x ALS products or to Version 4.x ASCII products.

Reference

Section III

Chapter

Utilities for Data Migration

11

This chapter contains information about tools that you can use to move data on Informix database server products.

The chapter contains syntax and use information for the following utilities:

- The **dbexport** utility unloads a database into text files for later import into another database.
- The **dbimport** utility creates and populates a database from text files.
- The **dbload** utility loads data into databases or tables that were created with Informix products.
- The **dbschema** utility prints SQL statements necessary to replicate a specified table, view, or database. It also shows the distributions that UPDATE STATISTICS creates.
- The **onload** utility loads data that was created with the **onunload** command into the database server.
- The **onunload** utility unloads data from an OnLine Dynamic Server or OnLine Workgroup Server database.
- The **onmode** utility modifies the data in an Informix database so that an earlier version of the database server can access it. (Other functions of **onmode** are documented in your *Administrator's Guide*.)

The chapter also contains brief descriptions of the following tools that are described in other manuals:

- The LOAD statement allows you to append rows to an existing table in a database. For complete documentation of the LOAD statement, refer to the *Informix Guide to SQL: Syntax*.
- The UNLOAD statement lets you write rows extracted by a SELECT statement to a file. For complete documentation of the LOAD statement, refer to the *Informix Guide to SQL: Syntax*.

AD/XP

XPS

- High-performance parallel loading with *external tables* in Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options and INFORMIX-OnLine XPS allows you to load and unload data. For complete documentation on external tables, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options*, or the *INFORMIX-OnLine XPS Administrator's Guide* and the *Informix Guide to SQL: Syntax*. ♦

WIN NT

- Informix Enterprise Command Center (IECC) provides a graphical user interface that allows you to load and unload single tables or entire databases to and from Informix database servers. For more information, refer to the *Informix Enterprise Command Center Installation Guide*. ♦

Setting Environment Variables

Before you use any of these migration utilities, you must set your **PATH**, **INFORMIXDIR**, and **INFORMIXSERVER** environment variables. For information about environment variables, see the *Informix Guide to SQL: Reference*.

Tip: *If you are using INFORMIX-SE, Informix recommends that you not use the* ***.dbs*** *directory as your current directory when you use a database-related utility. This practice keeps the* ***.dbs*** *directory free from file clutter and prevents multiple users from overwriting files that belong to other users.*

SE

For information about SE administration utilities, see the *INFORMIX-SE Administrator's Guide*. ♦

The dbexport Utility

The **dbexport** utility unloads a database into text files and creates a schema file. You can use the schema file with **dbimport** to re-create the database schema in another Informix environment. You can edit the schema file to modify the database that **dbimport** creates. The **dbexport** utility supports Universal Server data types.

AD/XP

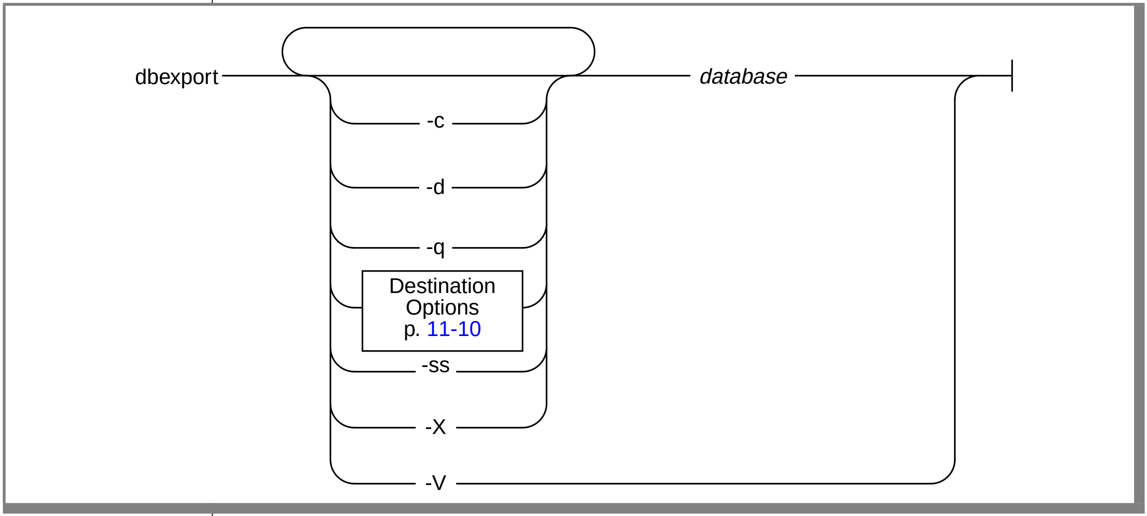
XPS

The **dbexport** and **dbimport** utilities are not part of OnLine XPS and Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options. Tables and schema need to be unloaded and loaded separately. ♦

The **dbexport** utility supports the following destination options:

- Unload a database and its schema file to disk.
- Unload a database and its schema file to tape.
- Unload the schema file to disk, and unload the data to tape.

Tip: *The examples in this chapter apply to all other Informix database servers unless they are marked specifically with an icon for INFORMIX-SE.*



Element	Purpose	Key Considerations
-c	Makes **dbexport** complete exporting unless a fatal error occurs.	**References**: For specific details on this option, see "Using the -c Option" on page 11-9.
-d	Makes **dbexport** export simple-large-object descriptors only, not simple-large-object data.	**References**: For more information about simple-large-object descriptors, refer to the *Guide to the Optical Subsystem*. **Restrictions**: Not supported by SE.
-q	Suppresses the display of error messages, warnings, and generated SQL data-definition statements.	None.
-ss	Generates database server-specific information for all tables in the specified database.	**References**: For specific details on this option, see "Using the -ss Option" on page 11-9.
-X	Recognizes HEX binary data in character fields.	None.
-V	Displays product version information.	None.
database	Specifies the name of the database that you want to export.	**Additional Information**: If your locale is set to use multibyte characters, you can use multibyte characters for the database name. **References:** If you want to use more than the simple name of the database, refer to the Database Name section of the *Informix Guide to SQL: Syntax*.

You must have DBA privilege or log in as user **informix** to export a database.

GLS

When the environment variables are set correctly, as described in the *Informix Guide to GLS Functionality*, **dbexport** can handle foreign characters in data and export the data from GLS databases. For more information, refer to "Changing the Database Name" on page 11-20. ♦

You can use delimited identifiers with the **dbexport** utility. The utility detects database objects that are keywords, mixed case, or have special characters and places double quotes around them.

In addition to the data files and the schema file, **dbexport** creates a file of messages called **dbexport.out** in the current directory. This file contains error messages, warnings, and a display of the SQL data definition statements that it generates. The same material is also written to the standard output unless you specify the **-q** option.

During the export, the database is locked in exclusive mode. If **dbexport** cannot obtain an exclusive lock, it displays a diagnostic message and exits.

You can press the INTERRUPT key at any time to cancel **dbexport**. The **dbexport** utility asks for confirmation before it terminates.

Using the -c Option

Even if you use the **-c** option, **dbimport** interrupts processing if one of the following fatal errors occurs:

- Unable to open the tape device specified
- Bad writes to the tape or disk
- Invalid command parameters
- Cannot open database or no system permission

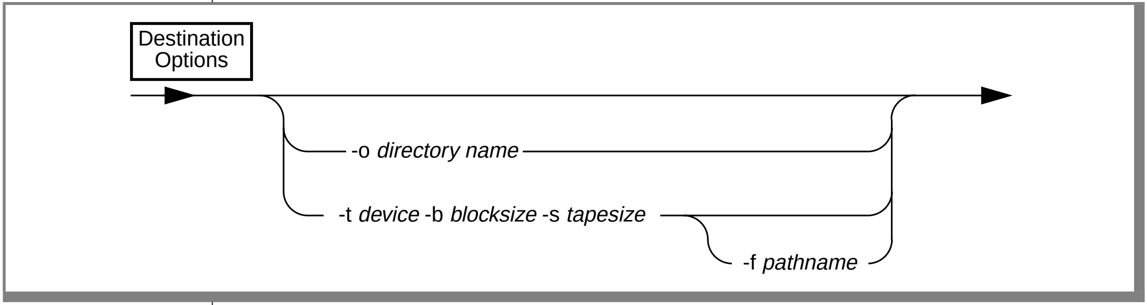
Using the -ss Option

The **-ss** option generates server-specific information. The **-ss** option specifies initial- and next-extent sizes, fragmentation information if the table is fragmented, the locking mode, the dbspace for a table, the blobspace for any simple large objects, and the dbspace for any smart large objects.

SE

For INFORMIX-SE databases, the **-ss** option generates the pathname of each table that is in a path other than the database directory. ♦

Destination Options



Element	Purpose	Key Considerations
-b *blocksize*	Specifies, in kilobytes, the block size of the tape device.	None.
-f *pathname*	Specifies the pathname where you want the schema file stored, if you are storing the data files on tape.	**Additional Information:** The pathname can be a complete pathname or simply a filename. If only a filename is given, the file is stored in the current directory.
-o *directory name*	Names the directory on disk in which **dbexport** creates the *database*.**exp** directory. This directory holds the data files and the schema file that **dbexport** creates for the *database*.	**Restrictions:** The directory specified as *directory name* must already exist.
-s *tapesize*	Specifies, in kilobytes, the amount of data that you can store on the tape.	**Restrictions:** The tape size is limited to 2,097,151 kilobytes. The limit is required because of the way **dbexport** and **dbimport** track their positions into the tape.
-t *device*	Specifies the pathname of the tape device where you want the text files and, possibly, the schema file stored.	**Restrictions:** The **-t** option does not allow you to specify a remote tape device.

When you write to disk, **dbexport** creates a subdirectory, *database*.**exp**, in the directory that the **-o** option specifies. The **dbexport** utility creates a file with the **.unl** extension for each table in the database. The schema file is written to the file *database*.**sql**. The **.unl** and **.sql** files are stored in the *database*.**exp** directory.

If you do not specify a destination for the data and schema files, the subdirectory *database*.**exp** is placed in the current working directory.

When you write the data files to tape, you can use the **-f** option to store the schema file to disk. You are not required to name the schema file *database*.**sql**. You can give it any name.

SE

The following **dbexport** command creates a **reports.exp** subdirectory in the current directory. It then unloads the **reports** database in the **turku** directory on the SE database server called **finland** and places the resulting files in the **reports.exp** directory.

```
dbexport //finland/turku/reports
```

♦

UNIX

For other non-SE database servers, the same command is as follows:

```
dbexport //finland/reports
```

The following command exports the database **stores7** to tape with a block size of 16 kilobytes and a tape capacity of 24,000 kilobytes. The schema file is written to **/tmp/stores7.imp**.

```
dbexport -t /dev/rmt0 -b 16 -s 24000 -f /tmp/stores7.imp
    stores7
```

The following command exports the same **stores7** database to the directory named **/work/exports/stores7.exp**. The resulting schema file is **/work/exports/stores7.exp/stores7.sql**.

```
dbexport -o /work/exports stores7
```

♦

WIN NT

For Windows NT, the following command exports the database **stores7** to tape with a block size of 16 kilobytes and a tape capacity of 24,000 kilobytes. The schema file is written to **C:\temp\stores7.imp**.

```
dbexport -t \\.\TAPEDRIVE -b 16 -s 24000 -f
    C:\temp\stores7.imp  stores7
```

The following command exports the same **stores7** database to the directory named **D:\work\exports\stores7.exp**. The resulting schema file is **D:\work\exports\stores7.exp\stores7.sql**.

```
dbexport -o D:\work\exports stores7
```

♦

The Contents of the Schema File

The schema file contains the SQL statements that you need to re-create the exported database. You can edit the schema file to modify the schema of the database.

IUS

UD

The schema file supports all Informix Dynamic Server with Universal Data Option and Universal Server data types. ♦

If you use the **-ss** option, the schema file contains server-specific information, such as initial- and next-extent sizes, fragmentation information, lock mode, the dbspace where each table resides, the blobspace where each simple-large-object column resides, and the dbspace for smart-large-objects. The following information is not retained:

- Logging mode of the database (For information about logging modes, refer to the *Informix Guide to SQL: Reference*.)
- The starting values of SERIAL columns

The statements in the schema file that create tables, views, indexes, roles, and grant privileges do so with the name of the user who originally created the database. In this way, the original owner retains DBA privileges for the database and is the owner of all the tables, indexes, and views. In addition, the person who executes the **dbimport** command also has DBA privileges for the database.

The schema file that **dbexport** creates contains comments, enclosed in curly braces, with information about the number of rows, columns, and indexes in tables, and information about the unload files. **dbimport** uses the information in these comments to load the database.

Warning: *Do NOT delete any comments in the schema file. Informix strongly recommends that you do not change any existing comments or add any new comments, or the* ***dbimport*** *may abort or produce unpredictable results.*

The number of rows should match in the unload file and the corresponding unload comment in the schema file. If you change the number of rows in the unload file but not the number of rows in the schema file, a mismatch occurs.

Tip: *If you delete some rows from an unload file, update the comment in the schema file with the correct number of rows for that unload file. Then the **dbimport** will be successful.*

IUS

UD

Exporting Simple Large Objects

When **dbimport**, **dbexport**, and DB-Access process blob data (simple large objects), they create temporary files for that data. Before you export or import data from tables that contain simple large objects, you must have one of the following items:

- A **\tmp** directory on your currently active drive
- The **DBTEMP** environment variable set to point to a directory that is available for temporary storage of the simple large objects

WIN NT

Windows NT sets the TMP and TEMP environment variables in the command prompt sessions, by default. However, if the TMP, TEMP, and DBTEMP environment variables are not set, **dbimport** places the temporary files for the simple large objects in the **\tmp** directory. ♦

Warning: *If a table has a CLOB or BLOB in a column, you cannot **dbexport** to a tape. If a table has a UDT in a column, **dbexport** to a tape may yield unpredictable results, depending on the export function of the UDT. Exported CLOB sizes are stored in hex format in the unload file.*

The dbimport Utility

The **dbimport** utility creates a database and loads it with data from text files. The input files consist of a schema file that is used to re-create the database and data files that contain the database data. Normally, you generate the input files with the **dbexport** utility, but you can use any properly formatted input files. The **dbimport** utility supports new data types used by Universal Server and Informix Dynamic Server with Universal Data Option.

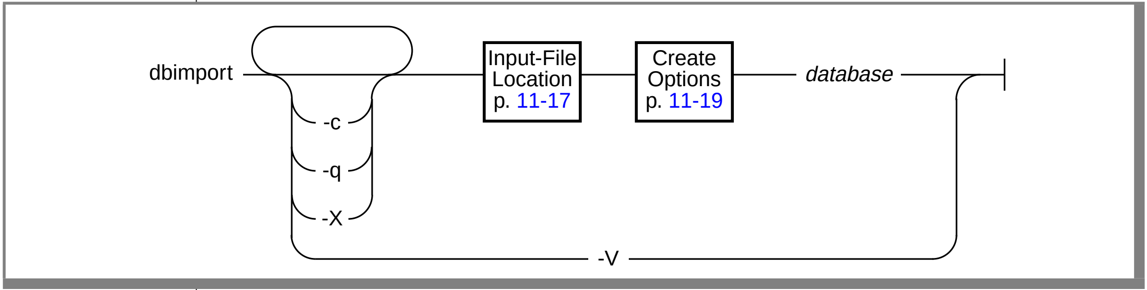
AD/XP

XPS

The **dbimport** command is not part of Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options or OnLine XPS.

Tables and schema need to be unloaded and loaded separately. ♦

Tip: *The examples in this chapter apply to all other Informix database servers, unless marked specifically with an icon for INFORMIX-SE.*



Element	Purpose	Key Considerations
-c	Instructs **dbimport** to complete importing even though it encounters certain nonfatal errors.	**References**: For more information, refer to "Using the -c Option" on page 11-16.
-q	Suppresses the display of error messages, warnings, and generated SQL data-definition statements.	None.

(1 of 2)

Element	Purpose	Key Considerations
-V	Displays product version information.	None.
-**X**	Recognizes HEX binary data in character fields.	None.
database	Specifies the name of the database to create.	**Additional Information**: If you want to use more than the simple name of the database, refer to the Database Name section of the *Informix Guide to SQL: Syntax*.

(2 of 2)

The **dbimport** utility can use files from the following location options:

- All input files are located on disk.
- All input files are located on tape.
- The schema file is located on disk, and the data files are located on tape.

Important: *Do not put comments into your input file. Comments may cause unpredictable results when the **dbimport** utility reads them.*

The **dbimport** utility supports the following options for a new Informix (excluding INFORMIX-SE) database server:

- Create an ANSI-compliant database (includes unbuffered logging).
- Establish transaction logging for a database (unbuffered or buffered logging).
- Specify the dbspace where the database will reside.

SE

The **dbimport** utility supports the following options for a new SE database:

- Create an ANSI-compliant database (ANSI-compliant logging).
- Establish transaction logging for a database (unbuffered logging). ♦

The user who runs **dbimport** is granted the DBA privilege on the newly created database. The **dbimport** process locks each table as it is being loaded and unlocks the table when the loading is completed.

GLS

When the GLS environment variables are set correctly, as described in the *Informix Guide to GLS Functionality*, **dbimport** can import data into database versions that support GLS. ♦

Using the -c Option

If you include the **-c** option, **dbimport** ignores the following errors:

- A data row that contains too many columns
- Inability to put a lock on a table
- Inability to release a lock

Even if you use the **-c** option, **dbimport** interrupts processing if one of the following fatal errors occurs:

- Unable to open the tape device specified
- Bad writes to the tape or disk
- Invalid command parameters
- Cannot open database or no system permission
- Cannot convert the data

The **dbimport** utility creates a file of messages called **dbimport.out** in the current directory. This file contains any error messages and warnings that are related to **dbimport** processing. The same information is also written to the standard output unless you specify the **-q** option.

Canceling dbimport

You can press the INTERRUPT key at any time to cancel **dbimport**. The **dbimport** program asks for confirmation before it terminates.

Database-Logging Mode

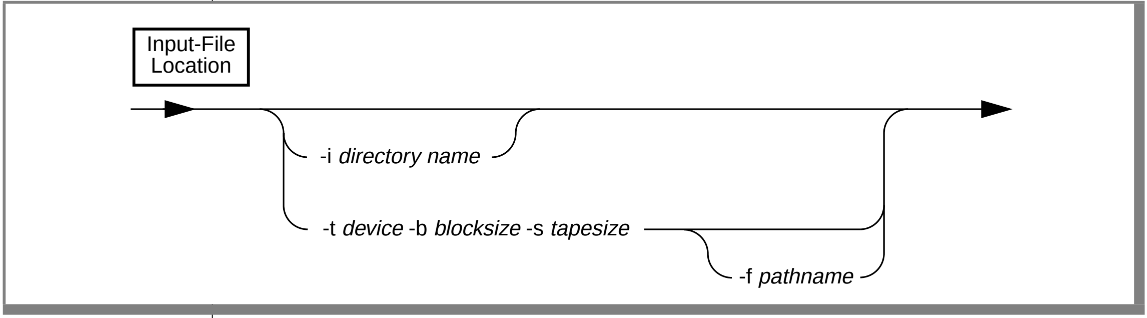
The logging mode is not retained in the schema file. You can specify any of the following options when you use **dbimport** to import a database:

- ANSI-compliant database with unbuffered logging
- Unbuffered logging
- Buffered logging

For more information, refer to "Create Options" on page 11-19.

Input-File Location Options

The input-file location tells **dbimport** where to look for the *database*.**exp** directory, which contains the files that **dbimport** will import. If you do not specify an input-file location, **dbimport** looks for data files in the directory *database***.exp** under the current directory and for the schema file in *database***.exp/***database***.sql**.



Element	Purpose	Key Considerations
-b *blocksize*	Specifies, in kilobytes, the block size of the tape device.	**Restrictions:** If you are importing from tape, you must use the same block size that you used to export the database.
-f *pathname*	Specifies where **dbimport** can find the schema file to use as input to create the database when the data files are read from tape.	**Additional Information:** If you use the **-f** option to export a database, you typically use the same pathname that you specified in the **dbexport** command. If you specify only a filename, **dbimport** looks for the file in the **.exp** subdirectory of your current directory.
-i *directory name*	Specifies the complete pathname on disk of the *database*.**exp** directory, which holds the input data files and schema file that **dbimport** uses to create and load the new database. The directory name should be the same as the database name.	**Additional Information:** This directory should be the same directory that you specified with the **dbexport -o** option. If you change the directory name, you also rename your database.

(1 of 2)

Element	Purpose	Key Considerations
-s *tapesize*	Specifies, in kilobytes, the amount of data that you can store on the tape.	**Restrictions:** If you are importing from tape, you must use the same tape size that you used to export the database.
-t *device*	Specifies the pathname of the tape device that holds the input files.	**Restrictions:** The **-t** option does *not* allow you to specify a remote tape device.

(2 of 2)

UNIX

The following command imports the **stores7** database from a tape with a block size of 16 kilobytes and a capacity of 24,000 kilobytes. The schema file is read from **/tmp/stores7.imp**.

```
dbimport -c  -t /dev/rmt0 -b 16 -s 24000 -f
    /tmp/stores7.imp stores7
```

The following command imports the **stores7** database from the **stores7.exp** directory under the **/work/exports** directory. The schema file is assumed to be **/work/exports/stores7.exp/stores7.sql**.

```
dbimport -c -i /work/exports stores7
```

♦

WIN NT

The following command imports the **stores7** database from a tape with a block size of 16 kilobytes and a capacity of 24,000 kilobytes. The schema file is read from **C:\temp\stores7.imp**.

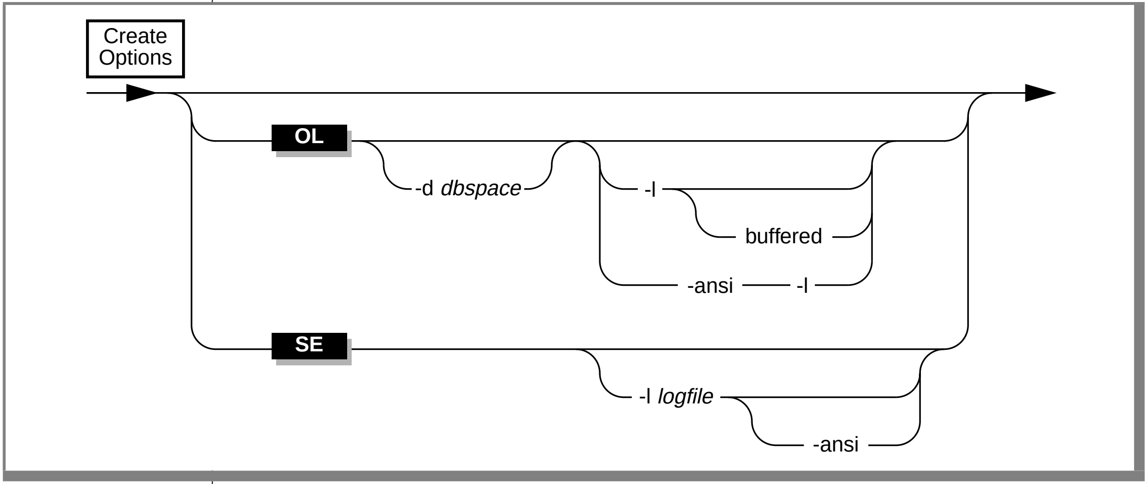
```
dbimport -c  -t \\.\TAPEDRIVE -b 16 -s 24000 -f
    C:\temp\stores7.imp stores7
```

The following command imports the **stores7** database from the **stores7.exp** directory under the **D:\work\exports** directory. The schema file is assumed to be **D:\work\exports\stores7.exp\stores7.sql**.

```
dbimport -c -i D:\work\exports stores7
```

♦

Create Options



Element	Purpose	Key Considerations
-ansi	Creates an ANSI-compliant database in which the ANSI rules for transaction logging are enabled.	**Additional Information:** If you specify the **-ansi** option, you must also specify the **-l** *logfile* option. For more information about ANSI-compliant databases, refer to the *Informix Guide to SQL: Reference*.
-d *dbspace*	Names the dbspace where the database is created. The default dbspace location is the rootdbs.	**Additional Information:** For SE, the database is always in the current directory.
-l	Establishes unbuffered transaction logging for the imported database.	**References:** For more information, refer to "Using the -l Options" on page 11-20.
-l buffered	Establishes buffered transaction logging for the imported database.	**References:** For more information, refer to "Using the -l Options" on page 11-20.
-l *logfile*	Establishes transaction logging for the imported database and specifies the name of the transaction-log file.	**Restrictions:** For SE, the *logfile* filename must be an absolute pathname or in the current directory. **References:** For more information, refer to "Using the -l Options" on page 11-20.

UNIX

The following command imports the **stores7** database from the **/usr/informix/port/stores7.exp** directory. The new database is ANSI compliant, and the transaction-log file is specified as **stores7.log** in **/usr/work**.

```
dbimport -c stores7 -i /usr/informix/port -l
    /usr/work/stores7.log -ansi
```

♦

WIN NT

The following command imports the **stores7** database from the **C:\USER\informix\port\stores7.exp** directory. The new database is ANSI compliant, and the transaction-log file is specified as **stores7.log** in **C:\USER\work**.

```
dbimport -c stores7 -i C:\USER\informix\port -l
    C:\USER\work\stores7.log -ansi
```

♦

Using the -l Options

The **-l** options are equivalent to the logging clauses of the CREATE DATABASE statement, as follows:

- The **-l** option is equivalent to the WITH LOG clause.
- The **-l buffered** option is equivalent to the WITH BUFFERED LOG.
- SE The **-l** *logfile* option is equivalent to the WITH LOG IN clause. ♦

For more information about the CREATE DATABASE statement, see the *Informix Guide to SQL: Syntax*.

Changing the Database Name

The **dbimport** utility assumes that the new database has the same name as the database that you exported. If you export a database to tape, you cannot change its name when you import it with **dbimport**.

If you export a database to disk, you can change the database name.

UNIX

To change the database name to newname on UNIX

In the following example, assume that **dbexport** unloaded the database **stores7** into the directory **/work/exports/stores7.exp**. Thus, the data files (the **.unl** files) are stored in **/work/exports/stores7.exp**, and the schema file is **/work/exports/stores7.exp/stores7.sql**.

1. Change the name of the **.exp** directory. That is, change **/work/exports/stores7.exp** to **/work/exports/newname.exp**.
2. Change the name of the schema file. That is, change **/work/exports/stores7.exp/stores7.sql** to **/work/exports/stores7.exp/newname.sql**. Do not change the names of the **.unl** files.
3. Import the database with the following command:

```
dbimport -i /work/exports newname
```

♦

WIN NT

To change the database name to newname on Windows NT

In the following example, assume that **dbexport** unloaded the database **stores7** into the directory **D:\work\exports\stores7.exp**. Thus, the data files (the **.unl** files) are stored in **D:\work\exports\stores7.exp**, and the schema file is **D:\work\exports\stores7.exp\stores7.sql**.

1. Change the name of the **.exp** directory. That is, change **D:\work\exports\stores7.exp** to **D:\work\exports\newname.exp**.
2. Change the name of the schema file. That is, change **D:\work\exports\stores7.exp\stores7.sql** to **D:\work\exports\stores7.exp\newname.sql**. Do not change the names of the **.unl** files.
3. Import the database with the following command:

```
dbimport -i D:\work\exports
```

♦

WIN NT

Importing Simple Large Objects and BYTE and TEXT data

When **dbimport**, **dbexport**, and DB-Access process simple large objects (blobs) and BYTE and TEXT data, they create temporary files for that data. Before you export or import data from tables that contain simple large objects or BYTE and TEXT data, you must have one of the following conditions:

- A **\tmp** directory on your currently active drive
- The **DBTEMP** environment variable set to point to a directory that is available for temporary storage of the simple large objects or BYTE and TEXT data

Windows NT sets the TMP and TEMP environment variables in the command prompt sessions, by default. However, if the TMP, TEMP, and DBTEMP environment variables are not set, **dbimport** places the temporary files for simple large objects or BYTE and TEXT data in the **\tmp** directory. ♦

Using dbimport to Change the Locale of a Database

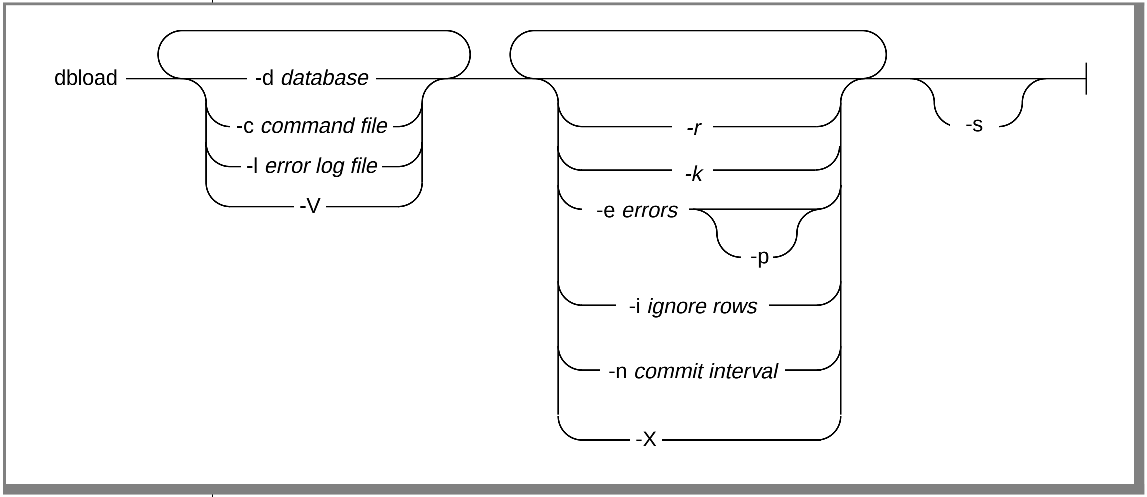
You can use **dbimport** to change the locale of a database.

To change the locale of a database

1. Set the **DB_LOCALE** environment variable to the name of the current database locale.
2. Run **dbexport** on the database.
3. Use the **DROP DATABASE** statement to drop the database that has the current locale name.
4. Set the **DB_LOCALE** environment variable to the desired database locale for the database.
5. Run **dbimport** to create a new database with the desired locale and import the data into this database.

The dbload Utility

The **dbload** utility transfers data from one or more text files into one or more existing tables. The **dbload** utility supports new data types used in Informix Dynamic Server with Universal Data Option and Universal Server.



Element	Purpose	Key Considerations
-c *command file*	Specifies the filename or pathname of a **dbload** command file.	**References:** For information about building the command file, refer to "Creating a dbload Command File" on page 11-28.
-d *database*	Specifies the name of the database to receive the data.	**Additional Information:** If you want to use more than the simple name of the database, refer to the Database Name section of the *Informix Guide to SQL: Syntax*.
-e *errors*	Specifies the number of bad rows that **dbload** reads before terminating. The default value for *errors* is 10.	**References:** For more information, refer to "Using the -e and -p Options" on page 11-27.
-i *ignore rows*	Specifies the number of rows to ignore in the input file.	**References:** For more information, refer to "Using the -i Option" on page 11-26.

(1 of 2)

Element	Purpose	Key Considerations
-k	Instructs **dbload** to lock the tables listed in the command file in exclusive mode during the load operation.	**References:** For more information, refer to "Using the -k Option" on page 11-26. **Restrictions:** You cannot use the **-k** option with the **-r** option because the **-r** option specifies that no tables are locked during the load operation.
-l *error log file*	Specifies the filename or pathname of an error log file.	**Restrictions:** If you specify an existing file, its contents are overwritten. If you specify a file that does not exist, **dbload** creates the file. **Additional Information:** The error log file stores diagnostic information and any input file rows that **dbload** cannot insert into the database.
-n *commit interval*	Specifies the commit interval in number of rows. The default interval is 100 rows.	**Additional Information:** If your database supports transactions, **dbload** commits a transaction after the specified number of new rows is read and inserted. A message appears after each commit. **References:** For information about transactions, see the *Informix Guide to SQL: Tutorial*.
-p	Prompts for instructions if the number of bad rows exceeds the limit.	**References:** For more information, refer to "Using the -e and -p Options" on page 11-27.
-r	Prevents **dbload** from locking the tables during a load, thus enabling other users to update data in the table during the load.	**Additional Information:** For more information, refer to "Using the -r Option" on page 11-26. **Restrictions:** You cannot use the **-r** option with the **-k** option because the **-r** option specifies that the tables are not locked during the load operation while the **-k** option specifies that the tables are locked in exclusive mode.
-s	Checks the syntax of the statements in the command file without inserting data.	**Additional Information:** The standard output displays the command file with any errors marked where they are found.
-V	Displays product version information.	
-X	Recognizes HEX binary data in character fields.	

(2 of 2)

Tip: *If you specify part (but not all) of the required information,* ***dbload*** *prompts you for additional specifications. The database name, command file, and error log file are all required. If you are missing all three options, you receive an error message.*

Using the dbload Utility

If you are on a network, include the database server name and directory path with the database name to specify a database on another database server or coserver.

You can load blobs with the **dbload** utility as long as the simple large objects are in text files.

The presence of indexes greatly affects the speed with which the **dbload** utility loads data. For best performance, drop any indexes on the tables that receive the data before you run **dbload**. You can create new indexes after **dbload** has finished.

You can use delimited identifiers with the **dbload** utility. The utility detects database objects that are keywords, mixed case, or have special characters, and places double quotes around them.

If your most-recent **dbload** session ended prematurely, specify the starting line number in the command-line syntax to resume loading with the next record in the file.

If you press the INTERRUPT key, **dbload** terminates and discards any new rows that were inserted but not yet committed to the database (if the database has transactions).

SE

Using dbload with SE

The following command loads data into the **stores7** database in the **turku** directory on the SE database server **finland**:

```
dbload -d //finland/turku/stores7 -c commands -l errlog
```

Using the -k Option

If you do not specify the **-k** option, the tables specified in the command file are locked in shared mode. When tables are locked in shared mode, the database server still has to acquire exclusive row or page locks when it inserts rows into the table.

When you specify the **-k** option, the database server places an exclusive lock on the entire table. The **-k** option increases performance for large loads because the database server does not have to acquire exclusive locks on rows or pages as it inserts rows during the load operation.

Using the -r Option

If you do not specify the **-r** option, the tables specified in the command file are locked during loading so that other users cannot update data in the table. Table locking reduces the number of locks needed during the load but reduces concurrency. If you are planning to load a large number of rows, use table locking and load during nonpeak hours.

To override this default lock mode, specify the **-k** option. The **-k** option instructs **dbload** to lock the tables in exclusive mode rather than shared mode during the load operation.

Using the -i Option

The **-i** option instructs **dbload** to read and ignore the specified number of new-line characters in the input file before it begins to process. This option is useful if your most-recent **dbload** session ended prematurely. For example, if **dbload** ends after it inserts 240 lines of input, you can begin to load again at line 241 if you set *number rows ignore* to `240`. It is also useful if header information in the input file precedes the data records.

Using the -e and -p Options

The **-e** option lets you specify how many bad rows to allow before **dbload** terminates.

If you set *errors* to a positive integer, **dbload** terminates when it reads (*errors* + 1) bad rows. If you set *errors* to zero, **dbload** terminates when it reads the first bad row.

If **dbload** exceeds the bad-row limit and the **-p** option is specified, **dbload** prompts you for instructions before it terminates. The prompt asks whether you want to roll back or to commit all rows that were inserted since the last transaction.

If **dbload** exceeds the bad-row limit and the **-p** option is not specified, **dbload** commits all rows that were inserted since the last transaction.

Choosing Between dbload and LOAD

The **dbload** utility offers the following advantages over the LOAD statement:

- You can use **dbload** to load data from input files that were created with a variety of format arrangements. The **dbload** command file can accommodate data from entirely different database management systems.
- You can specify a starting point in the load by directing **dbload** to read but ignore *x* number of rows.
- You can specify a batch size so that after every *x* number of rows are inserted, the insert is committed.
- You can limit the number of bad rows read, beyond which **dbload** ends.

The cost of **dbload** flexibility is the time and effort spent creating the **dbload** command file, which is required for **dbload** operation. The input files are not specified as part of the **dbload** command line, and neither are the tables into which the data is inserted. This information is contained in the command file.

Creating a dbload Command File

Before you use **dbload**, you must create a command file that names the input data files and the tables that receive the data. The command file maps fields from one or more input files into columns of one or more tables within your database.

The command file contains only FILE and INSERT statements. Each FILE statement names an input data file. The FILE statement also defines the data fields from the input file that are inserted into the table. Each INSERT statement names a table to receive the data. The INSERT statement also defines how **dbload** places the data that is described in the FILE statement into the table columns.

Within the command file, the FILE statement can appear in the following forms:

- Delimiter form
- Character-position form

The FILE statement has a size limit of 4,096 bytes.

Use the delimiter form of the FILE statement when every field in the input data row uses the same delimiter and every row ends with a new-line character. This format is typical of data rows with variable-length fields. You can also use the delimiter form of the FILE statement with fixed-length fields as long as the data rows meet the delimiter and new-line requirements. The delimiter form of the FILE and INSERT statements is easier to use than the character-position form.

Use the character-position form of the FILE statement when you cannot rely on delimiters and you need to identify the input data fields by character position within the input row. For example, use this form to indicate that the first input data field begins at character position `1` and continues until character position `20`. You can also use this form if you must translate a character string into a null value. For example, if your input data file uses a sequence of blanks to indicate a null value, you must use this form if you want to instruct **dbload** to substitute null at every occurrence of the blank-character string.

You can use both forms of the FILE statement in a single command file. However, for clarity, the two forms are described separately in the following sections.

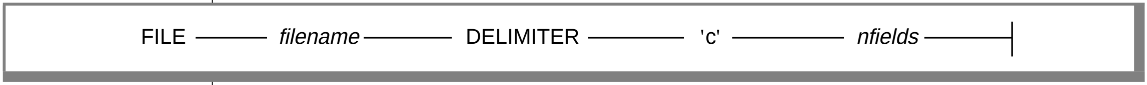
FILE and INSERT Statements: Delimiter Form

The following example of a **dbload** command file illustrates a simple delimiter form of the FILE and INSERT statements. The example is based on the **stores7** database. An UNLOAD statement created the three input data files, **stock.unl**, **customer.unl**, and **manufact.unl**. To see the **.unl** input data files, refer to the directory **$INFORMIXDIR/demo/***prod_name* (UNIX) or **%INFORMIXDIR%\demo\***prod_name* (Windows NT).

```
FILE stock.unl DELIMITER '|' 6;
INSERT INTO stock;
FILE customer.unl DELIMITER '|' 10;
INSERT INTO customer;
FILE manufact.unl DELIMITER '|' 3;
INSERT INTO manufact;
```

Syntax for the Delimiter Form

The following diagram shows the syntax of the delimiter FILE statement.



Element	Purpose	Key Considerations
c	Defines the field delimiter for the specific input file.	**Restrictions**: If the delimiter specified by **c** appears as a literal character anywhere in the input file, the character must be preceded with a backslash (\) in the input file. For example, if the value of **c** is specified as a square bracket ([), you must place a backslash before any literal square bracket that appears in the input file. Similarly, you must precede any backslash that appears in the input file with an additional backslash.
filename	Specifies the input file.	None.
nfields	Indicates the number of fields in each data row.	None.

The **dbload** utility assigns the sequential names **f01**, **f02**, **f03**, and so on to fields in the input file. You cannot see these names, but if you refer to these fields to specify a value list in an associated INSERT statement, you must use the **f01**, **f02**, **f03** format. For details, refer to "How to Write a dbload Command File in Delimiter Form" on page 11-31.

Two consecutive delimiters define a null field. As a precaution, you can place a delimiter immediately before the new-line character that marks the end of each data row. If the last field of a data row has data, you must use a delimiter. If you omit this delimiter, an error results whenever the last field of a data row is not empty.

Inserted data types correspond to the explicit or default column list. If the data field width is different from its corresponding character column width, the data is made to fit. That is, inserted values are padded with blanks if the data is not wide enough for the column or truncated if the data is too wide for the column.

If the number of columns named is fewer than the number of columns in the table, **dbload** inserts the default value that was specified when the table was created for the unnamed columns. If no default value is specified, **dbload** attempts to insert a null value. If the attempt violates a not null restriction or a unique constraint, the insert fails, and an error message is returned.

If the INSERT statement omits the column names, the default INSERT specification is every column in the named table. If the INSERT statement omits the VALUES clause, the default INSERT specification is every field of the previous FILE statement.

An error results if the number of column names listed (or implied by default) does not match the number of values listed (or implied by default).

The syntax of **dbload** INSERT statements resembles INSERT statements in SQL, except that in **dbload**, INSERT statements cannot incorporate SELECT statements.

Warning: *Do not use the CURRENT, TODAY, and USER keywords of the INSERT INTO statement in a **dbload** command file; they are not supported in the **dbload** command file. These keywords are supported in SQL only.*

For example, the following **dbload** command is not supported:

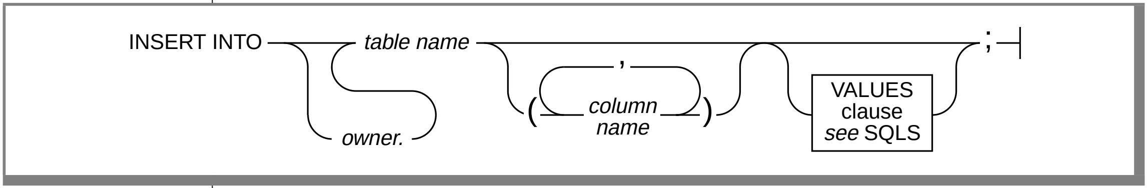
```
FILE "testtbl2.unl" DELIMITER '|' 1;
            INSERT INTO testtbl
                 (testuser, testtime, testfield)
              VALUES
                 ('kae', CURRENT, f01);
```

First, load the existing data, then write an SQL query to insert or update the data with the current time, date, or user login. You could write the following SQL statement:

```
INSERT INTO testtbl
               (testuser, testtime, testfield)
            VALUES
               ('kae', CURRENT, f01);
```

The CURRENT keyword returns the system date and time. The TODAY keyword returns the system date. The USER keyword returns the user login name.

The following diagram shows the syntax of the **dbload** INSERT statement for delimiter form.



Element	Purpose	Key Considerations
column name	Specifies the column that receives the new data.	None.
owner.	Specifies the user name of the table owner.	None.
table name	Specifies the table that receives the new data.	None.

Users who execute **dbload** with this command file must have the Insert privilege on the named table.

How to Write a dbload Command File in Delimiter Form

The first FILE AND INSERT statement set in the delimiter example on page 11-29 is repeated in the following example:

```
FILE stock.unl DELIMITER '|' 6;
INSERT INTO stock;
```

The FILE statement describes the **stock.unl** data rows as composed of six fields, each separated by a vertical bar (|) as the delimiter. Two consecutive delimiters define a null field. As a precaution, you can place a delimiter immediately before the new line character that marks the end of each data row. If the last field of a data row has data, you must use a delimiter. If you omit this delimiter, an error results.

Compare the FILE statement with the data rows in the following example, which appear in the input file **stock.unl**. (Because the last field is not followed by a delimiter, an error results if any data row ends with an empty field.)

```
1|SMT|baseball gloves|450.00|case|10 gloves/case
2|HRO|baseball|126.00|case|24/case
3|SHK|baseball bat|240.00|case|12/case
```

The example INSERT statement contains only the required elements. Because the column list is omitted, the INSERT statement implies that values are to be inserted into every field in the **stock** table. Because the VALUES clause is omitted, the INSERT statement implies that the input values for every field are defined in the most-recent FILE statement. This INSERT statement is valid because the **stock** table contains six fields, which is the same number of values that the FILE statement defines. The following example shows the first data row that is inserted into **stock** from this INSERT statement.

Field	Column	Value
f01	stock_num	1
f02	manu_code	SMT
f03	description	baseball gloves
f04	unit_price	450.00
f05	unit	case
f06	unit_descr	10 gloves/case

The FILE and INSERT statement in the following example illustrates a more complex INSERT statement syntax:

```
FILE stock.unl DELIMITER '|' 6;
INSERT INTO new_stock (col1, col2, col3, col5, col6)
    VALUES (f01, f03, f02, f05, 'autographed');
```

In this example, the VALUES clause uses the field names that **dbload** assigns automatically. You must reference the automatically assigned field names with the letter **f** followed by a number: **f01**, **f02**, **f10**, **f100**, **f999**, **f1000**, and so on. All other formats are incorrect.

Tip: *The first nine fields must include a zero: f01, f02, ..., f09.*

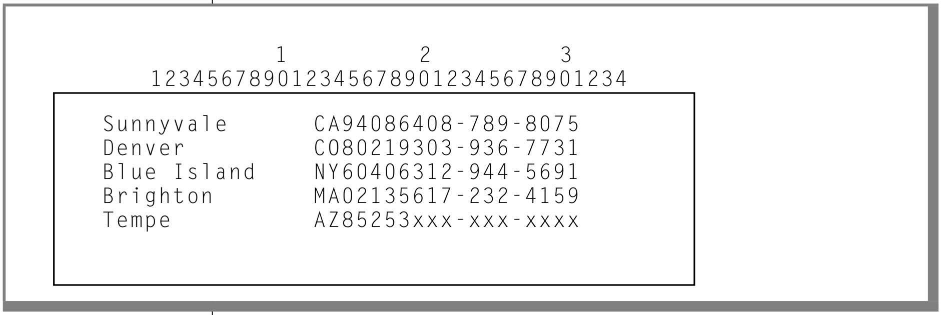
The user changed the column names, the order of the data, and the meaning of **col6** in the new **stock** table. Because the fourth column in **new_stock** (**col4**) is not named in the column list, the new data row contains a null in the **col4** position (assuming that the column permits nulls). If no default is specified for **col4**, the inserted value is null.

The following table shows the first data row that is inserted into **new_stock** from this INSERT statement.

Column	Value
col1	1
col2	baseball gloves
col3	SMT
col4	null
col5	case
col6	autographed

FILE and INSERT Statements: Character-Position Form

The examples in this section are based on an input data file, **cust_loc_data**, that contains the last four columns (**city**, **state**, **zipcode**, and **phone**) of the **customer** table. Fields in the input file are padded with blanks to create data rows in which the location of data fields and the number of characters are the same across all rows. The definitions for these fields are CHAR(15), CHAR(2), CHAR(5), and CHAR(12), respectively. Figure 11-1 displays the character positions and five example data rows from the **cust_loc_data** file.



```
                1         2         3
       1234567890123456789012345678901234
    Sunnyvale      CA94086408-789-8075
    Denver         CO80219303-936-7731
    Blue Island    NY60406312-944-5691
    Brighton       MA02135617-232-4159
    Tempe          AZ85253xxx-xxx-xxxx
```

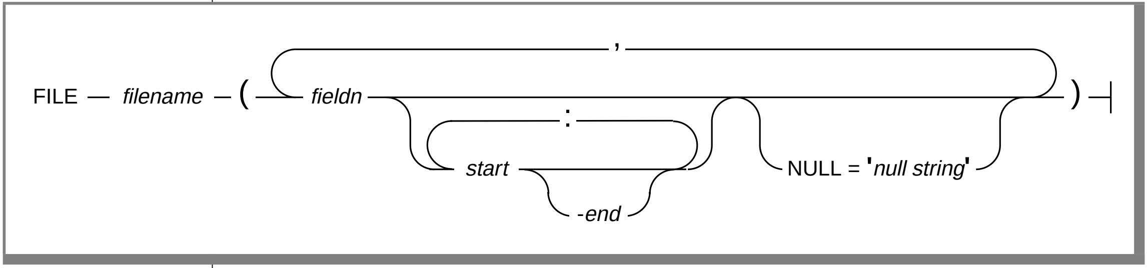
Figure 11-1
A Sample Data File

The following example of a **dbload** command file illustrates the character-position form of the FILE and INSERT statements. The example includes two new tables, **cust_address** and **cust_sort**, to receive the data. For the purpose of this example, **cust_address** contains four columns, the second of which is omitted from the column list. The **cust_sort** table contains two columns:

```
FILE cust_loc_data
    (city 1-15,
     state 16-17,
     area_cd 23-25 NULL = 'xxx',
     phone 23-34 NULL = 'xxx-xxx-xxxx',
     zip 18-22,
     state_area 16-17 : 23-25);
INSERT INTO cust_address (col1, col3, col4)
    VALUES (city, state, zip);
INSERT INTO cust_sort
    VALUES (area_cd, zip);
```

Syntax for the Character-Position Form

The following diagram shows the syntax of the character-position FILE statement.



Element	Purpose	Key Considerations
-end	Indicates the character position within a data row that ends a range of character positions.	**Restrictions**: A hyphen must precede the end value.
fieldn	Assigns a name to the data field that you are defining with the range of character positions.	None.
filename	Specifies the name of the input file.	None.
null string	Specifies the data value for which **dbload** should substitute a null.	**Restrictions**: Must be a quoted string.
start	Indicates the character position within a data row that starts a range of character positions. If you use *start* without *end*, it represents a single character.	None.

You can repeat the same character position in a data-field definition or in different fields.

The *null string* scope of reference is the data field for which you define it. You can define an explicit null string for each field that allows null entries.

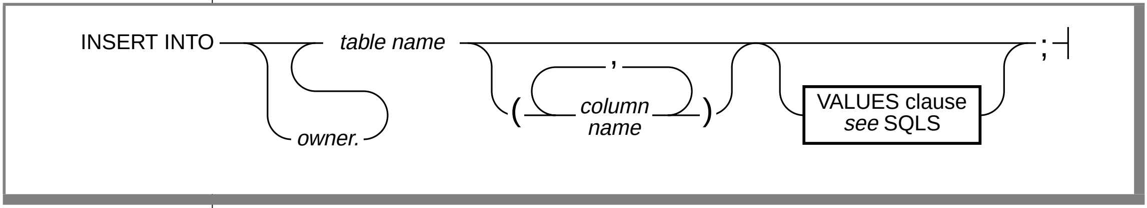
Inserted data types correspond to the explicit or default column list. If the data-field width is different from its corresponding character column, inserted values are padded with blanks if the column is wider or are truncated if the field is wider.

If the number of columns named is fewer than the number of columns in the table, **dbload** inserts the default value that is specified for the unnamed columns. If no default value is specified, **dbload** attempts to insert a null value. If the attempt violates a not null restriction or a unique constraint, the insert fails, and an error message is returned.

If the INSERT statement omits the column names, the default INSERT specification is every column in the named table. If the INSERT statement omits the VALUES clause, the default INSERT specification is every field of the previous FILE statement.

An error results if the number of column names listed (or implied by default) does not match the number of values listed (or implied by default).

The syntax of **dbload** INSERT statements resembles INSERT statements in SQL, except that in **dbload**, INSERT statements cannot incorporate SELECT statements. The following diagram shows the syntax of the **dbload** INSERT statement for character-position form.



Element	Purpose	Key Considerations
column name	Specifies the column that receives the new data.	None.
table name	Specifies the table that receives the new data.	None.
*owner***.**	Specifies the user name of the table owner.	None.

The syntax for character-position form is identical to the syntax for delimiter form.

The user who executes **dbload** with this command file must have the Insert privilege on the named table.

SE

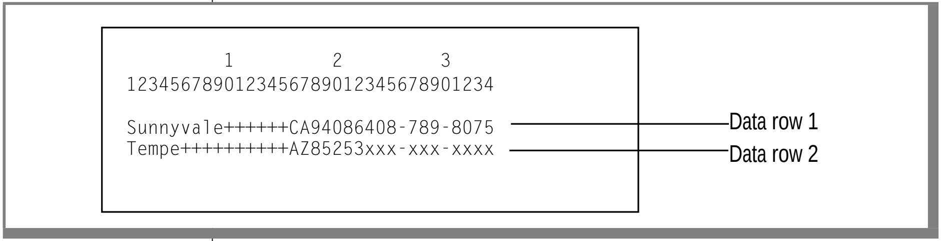
In INFORMIX-SE, the **dbload** utility recognizes valid SE table references, including owner designations. That is, the owner name can precede the table name but the database server name or the database name cannot precede the table name. Valid table-name syntax is defined in detail in the *Informix Guide to SQL: Syntax*. ♦

How to Write a dbload Command File in Character-Position Form

The first FILE AND INSERT statement set in the character-position example on page 11-34 is repeated in the following example:

```
FILE cust_loc_data
    (city 1-15,
     state 16-17,
     area_cd 23-25 NULL = 'xxx',
     phone 23-34 NULL = 'xxx-xxx-xxxx',
     zip 18-22,
     state_area 16-17 : 23-25);
INSERT INTO cust_address (col1, col3, col4)
    VALUES (city, state, zip);
```

The FILE statement defines six data fields from the **cust_loc_data** table data rows. The statement names the fields and uses character positions to define the length of each field. Compare the FILE statement in the preceding example with the data rows in Figure 11-2.



```
         1         2         3
1234567890123456789012345678901234

Sunnyvale++++++CA94086408-789-8075
Tempe++++++++++AZ85253xxx-xxx-xxxx
```

Data row 1
Data row 2

Figure 11-2
A Sample Data File

The FILE statement defines the following data fields, which are derived from the data rows in Figure 11-2 on page 11-37.

Column	Values from Data Row 1	Values from Data Row 2
city	Sunnyvale++++++	Tempe++++++++++
state	CA	AZ
area_cd	408	null
phone	408-789-8075	null
zip	94086	85253
state_area	CA408	AZxxx

The null strings that are defined for the **phone** and **area_cd** fields generate the null values in those columns but do not affect the values that are stored in the **state_area** column.

The INSERT statement uses the field names and values that are derived from the FILE statement as the value-list input. Consider the following INSERT statement:

```
INSERT INTO cust_address (col1, col3, col4)
    VALUES (city, state, zip);
```

The INSERT statement uses the data in Figure 11-2 and the FILE statement on page 11-37 to put the following information into the **cust_address** table.

Column	Values from Data Row 1	Values from Data Row 2
col1	Sunnyvale++++++	Tempe++++++++++
col2	null	null
col3	CA	AZ
col4	94086	85253

Because the second column (**col2**) in **cust_address** is not named, the new data row contains a null (assuming that the column permits nulls).

Consider the following INSERT statement:

```
INSERT INTO cust_sort
    VALUES (area_cd, zip);
```

This INSERT statement inserts the following data rows into the **cust_sort** table.

Column	Values from Data Row 1	Values from Data Row 2
col1	408	NULL
col2	94086	85253

Because no column list is provided, **dbload** reads the names of all the columns in **cust_sort** from the system catalog. (You cannot insert data into a temporary table because temporary tables are not entered into the system catalog.) Field names from the previous FILE statement specify the values to load into each column. You do not need one FILE statement for each INSERT statement.

IUS

UD

Writing a dbload Command File for Complex Data Types

This section describes how to write **dbload** command files that load columns that contain complex data types into tables. The examples cover how to use **dbload** with named row types, unnamed row types, sets, and lists.

Using dbload with Named Row Types

The procedure for how to use **dbload** with named row types is somewhat different than for other complex data types because named row types are actually user-defined data types. In fact, you can follow these steps for any user-defined data type.

This example uses a table **person** that contains one column with a named row type. The **person_t** named row type contains six fields: **name**, **address**, **city**, **state**, **zip**, and **bdate**.

The following syntax shows how to create the named row type and the table used in this example:

```
        CREATE ROW TYPE person_t
        (
            name VARCHAR(30) NOT NULL,
            address VARCHAR(20),
            city VARCHAR(20),
            state CHAR(2),
            zip VARCHAR(9),
            bdate DATE
        );
    CREATE TABLE person of TYPE person_t;
```

To load the data, follow these steps:

1. Use the UNLOAD statement to unload the table to an input file. In this example, the input file sees the named row type as six separate fields:

   ```
   Brown, James|13 First St.|San Francisco|CA|94070|01/04/1940|
   Karen Smith|5820 Easy Ave #100|Fremont|CA|94502|01/13/1983|
   ```

2. Use the **dbschema** utility to capture the schema of the table and the row type. You must use the **dbschema -u** option to pick up the named row type:

   ```
   dbschema -d stores7 -u person_t > schema.sql
   dbschema -d stores7 -t person > schema.sql
   ```

3. Use DB-Access to re-create the **person** table in the new database.

   For the detailed steps, see "Using DB-Access with dbschema Output" on page 11-60.

4. Create the **dbload** command file. This **dbload** command file inserts two rows into the **person** table in the new database.

   ```
   FILE person.unl DELIMITER '|' 6;
   INSERT INTO person;
   ```

   This **dbload** example shows how to insert new data rows into the **person** table. The number of rows in the INSERT statement and the **dbload** command file must match:

   ```
   FILE person.unl DELIMITER '|' 6;
       INSERT INTO person
       VALUES ('Jones, Richard', '95 East Ave.', 'Philadelphia', 'PA',
       '19115',
       '03/15/97');
   ```

5. Execute the **dbload** command:

   ```
   dbload -d newdb -c uds_command -l errlog
   ```

Tip: *To find the number of fields in an unloaded table containing a named row type, count the number of fields between each vertical bar (|) delimiter.*

Using dbload with Unnamed Row Types

You can use **dbload** with unnamed row types. In this example, the **devtest** table contains two columns with unnamed row types, **s_name** and **s_address**. The **s_name** column contains three fields: **f_name**, **m_init**, and **l_name**. The **s_address** column contains four fields: **street**, **city**, **state**, and **zip**:

```
CREATE TABLE devtest
     (
     s_name ROW(f_name varchar(20), m_init char(1), l_name
varchar(20) not null),
     s_address ROW(street varchar(20), city varchar(20), state
char(20), zip varchar(9)
     );
```

The data from the **devtest** table is unloaded into the **devtest.unl** file. Each data row contains two delimited fields, one for each unnamed row type. The ROW constructor precedes each unnamed row type:

```
ROW('Jim','K','Johnson')|ROW('10 Grove St.','Eldorado','CA','94108')|
ROW('Candy','S','Cane')|ROW('7 Willy Wonka
Ave.','Hershey','PA','17033')|
```

This **dbload** example shows how to insert data containing unnamed row types into the **devtest** table. Put double quotes around each unnamed row type or the insert will not work.

```
FILE devtest.unl DELIMITER '|' 2;
   INSERT INTO devtest (s_name, s_address)
   VALUES ("row('Craig', 'X', 'Smith')",
      "row('1200 Cheese Ave.', 'Rainy City', 'OR', '97200')");
```

Using dbload with Collection Data Types

You can use **dbload** with collection data types such as SET, LIST, and MULTISET.

SET Data Type Example

In a SET, each element is unique, and no nulls are allowed. The numbers of elements in a SET can vary. The following statement creates a table in which the **children** column is defined as a SET:

```
CREATE TABLE employee
    (
        name char(30),
        address char(40),
        children SET (varchar(30) NOT NULL)
    );
```

The data from the **employee** table is unloaded into the **employee.unl** file. Each data row contains four delimited fields. The first set contains three elements (**Karen**, **Lauren**, and **Andrea**) while the second set contains four elements. The SET constructor precedes each SET data row:

```
Muriel|5555 SW Merry
SailingDr.|02/06/1926|SET{'Karen','Lauren','Andrea'}|
   Larry|1234 Indian Lane|07/31/1927|SET{'Martha','Melissa','Craig','Larry'}|
```

This **dbload** example shows how to insert data containing SET data types into the **employee** table in the new database. Put double quotes around each SET data type or the insert does not work.

```
FILE employee.unl DELIMITER '|' 4;
INSERT INTO employee
VALUES ('Marvin', '10734 Pardee', '06/17/27', "SET{'Joe', 'Ann'}");
```

LIST Data Type Example

A list is an ordered collection of elements that allows duplicate values. The following statement creates a table in which the **month_sales** column is defined as a LIST:

```
CREATE TABLE sales_person
    (
        name CHAR(30),
        month_sales LIST(MONEY NOT NULL)
    );
```

The data from the **sales_person** table is unloaded into the **sales.unl** file. Each data row contains two delimited fields.

```
Jane Doe|LIST{'4.00','20.45','000.99'}|
Big Earner|LIST{'0000.00','00000.00','999.99'}|
```

This **dbload** example shows how to insert data containing LIST data types into the **sales_person** table in the new database. Put double quotes around each LIST data type or else the insert does not work:

```
FILE sales_person.unl DELIMITER '|' 2;
INSERT INTO sales_person
VALUES ('Jenny Chow', ”list{587900, 600000}”);
```

You can load multisets in a similar manner.

What Else Works with dbload

You can use **dbload** with the following data types:

- A BLOB or CLOB
- A SET inside a ROW type

What Does Not Work with dbload

The **dbload** utility does not work with the following data types:

- A CLOB or BLOB inside a ROW type
- A ROW type inside a SET

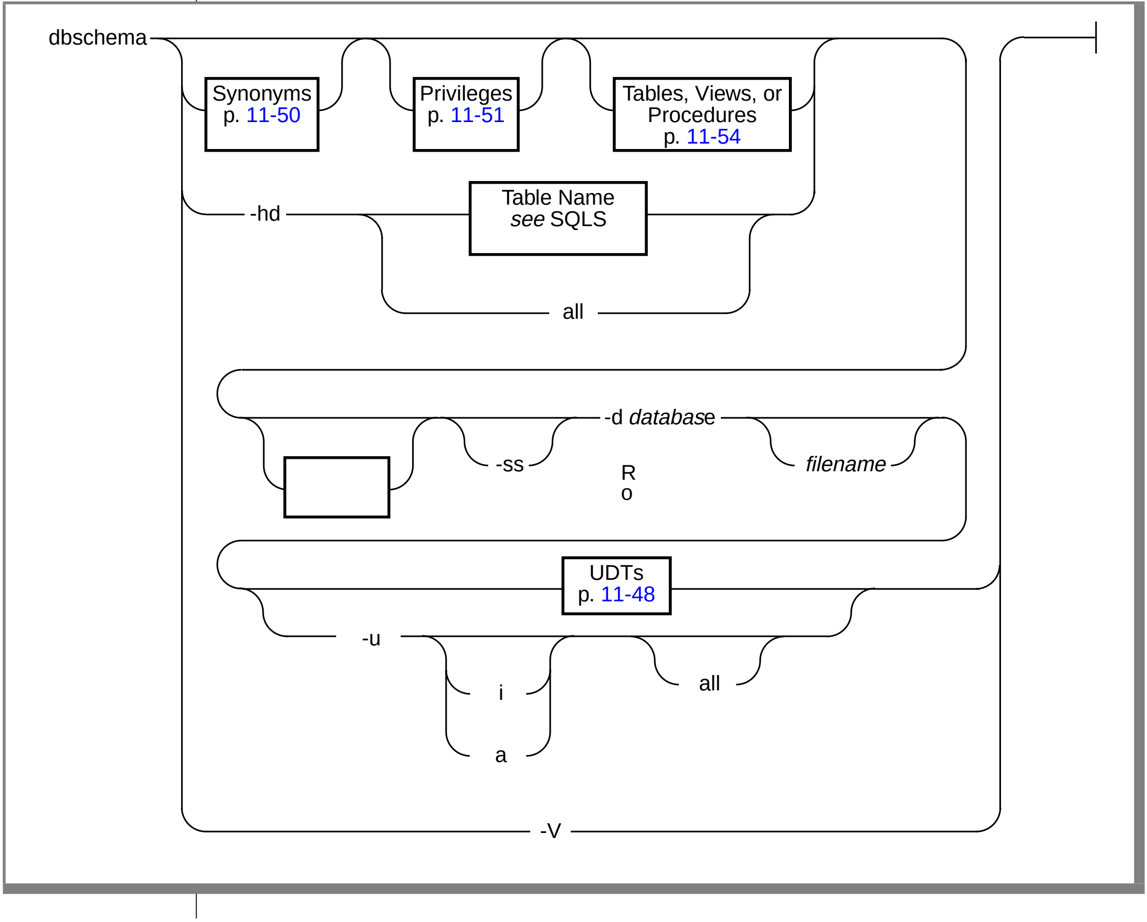
Warning: *All the load utilities (**dbexport**, **dbimport**, **load**, **unload**, and **dbload**) rely on an export and import function. If you do not define this function when you write a user-defined type, you cannot use these utilities.*

Loading a new data type inside another data type can cause problems if the representation of the data contains handles. If a string represents the data, you should be able to load it.

The dbschema Utility

You can use the **dbschema** utility for the following purposes:

- To display the SQL statements (the *schema*) that are required to replicate a database or a specific table, view, or procedure
- To display the schema for the Information Schema views
- To display the distribution information that is stored for one or more tables in the database
- To display information on user-defined data types and row types.



Element	Purpose	Key Considerations
-d *database/co-server*	Specifies the database or co-server to which the schema applies. The database can be on a remote database server.	**References:** If you want to use more than the simple name of the database, refer to the Database Name section of the *Informix Guide to SQL: Syntax*.
-hd	Displays the distribution as data values.	**References:** For more information, refer to "Displaying the Distribution Information for Tables" on page 11-56.
-ss	Generates server-specific information.	**Restrictions:** This option is ignored if no table schema is generated. **References:** For more information, refer to "Using the -ss Option" on page 11-48.
-u	Prints user-defined data types.	**References:** For more information, refer to "Using the -u Option" on page 11-48.
-ui	Prints user-defined data types, including type inheritance.	**References:** For more information, refer to "Using the -u Option" on page 11-48.
-ua	Prints user-defined data types including all functions and casts defined over a type.	**References:** For more information, refer to "Using the -u Option" on page 11-48.
-V	Displays product version information.	None.
all	Directs **dbschema** to include all the tables in the database in the display of distributions.	None.
filename	Specifies the filename to contain the **dbschema** output.	**Additional Information:** If you do not supply a *filename*, **dbschema** sends output to the screen. If you do supply a *filename*, **dbschema** creates a file to contain the **dbschema** output and gives it the name you specify.

You must be the DBA or have the Connect or Resource privilege to the database before you can run **dbschema** on it.

AD/XP

XPS

For Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options and OnLine XPS, you need to edit your schema files to incorporate the dbslices feature and the syntax definition of dbspaces. The **dbschema** utility in Dynamic Server with AD and XP Options and OnLine XPS produces dbslice information. For information on these schema issues, see Chapter 3, "Migrating OnLine Dynamic Server 7.2 to AD/XP 8.2." ♦

SE

If you are using INFORMIX-SE, the database must exist in your current directory or in a directory that is cited in your **DBPATH** environment variable. ♦

GLS

When the GLS environment variables are set correctly, as described in the *Guide to GLS Functionality*, **dbschema** can handle foreign characters and Version 7.2x and Universal Server databases. ♦

You can use delimited identifiers with the **dbschema** utility. The utility detects database objects that are keywords, mixed case, or have special characters and places double quotes around them.

Creating the Schema for a Database

You can create the schema for an entire database or for a portion of the database. The options for **dbschema** allow you to perform the following actions:

- Display CREATE SYNONYM statements by owner, for a specific table or for the entire database.
- Display the CREATE TABLE, CREATE VIEW, CREATE FUNCTION, or CREATE PROCEDURE statements for a specific table or for the entire database.
- Display all GRANT privilege statements that affect a specified user or that affect all users for a database or a specific table. The user can be either a user name or role name.

IUS

UD

- Display user-defined and row data types with or without type inheritance and casts. ♦

When you use **dbschema** and specify only the database name, it is equivalent to using **dbschema** with all its options (except for the **-hd** and **-ss** options). In addition, if Information Schema views were created for the database, this schema is shown. For example, the following two commands are equivalent:

```
dbschema -d stores7
dbschema -s all -p all -t all -f all -d stores7
```

The SERIAL fields included in CREATE TABLE statements that **dbschema** displays do not specify a starting value. New SERIAL fields created with the schema file have a starting value of 1, regardless of their starting value in the original database. If this value is not acceptable, you must modify the schema file.

Creating Schemas for Databases Across a Network

UNIX

You can specify a database on any accessible non-SE Informix database servers with the **-d** database syntax. The following command displays the schema for the **stores7** database on the **finland** database server on the UNIX system console:

```
dbschema -d //finland/stores7
```

♦

SE

To specify a database on another SE database server, include the database server name and directory path with the database name. The command in the following example displays the schema for the **stores7** database in the **turku** directory on the **finland** database server on the system console:

```
dbschema -d //finland/turku/stores7
```

♦

Owner Naming with dbschema

The **dbschema** utility uses the *owner.object* convention when it generates any CREATE TABLE, CREATE INDEX, CREATE SYNONYM, CREATE VIEW, CREATE PROCEDURE, CREATE FUNCTION, or GRANT statements, and when it reproduces any unique, referential, or check constraints. As a result, if you use the **dbschema** output to create a new object (table, index, view, procedure, constraint, or synonym), the owner of the original object owns the new object. If you want to change the owner of the new object, you must edit the **dbschema** output before you run it as an SQL script.

You can use the output of **dbschema** to create a new function if you also specify the *pathname* to a file in which compile-time warnings are stored. This pathname is displayed in the **dbschema** output.

For more information about the CREATE TABLE, CREATE INDEX, CREATE SYNONYM, CREATE VIEW, CREATE PROCEDURE, CREATE FUNCTION, and GRANT statements, see the *Informix Guide to SQL: Syntax*.

Using the -ss Option

The **-ss** option generates server-specific information. In all Informix database servers except SE, the **-ss** option always generates the lock mode, extent sizes, and the dbspace name if the dbspace name is different from the database dbspace. In addition, if tables are fragmented, the **-ss** option displays information about the fragmentation strategy.

When you specify the **dbschema -ss** option, the output also displays any GRANT FRAGMENT statements that are issued for a particular user or in the entire schema.

Important: *Use the **dbschema -ss** option to obtain information specific to a database server, including fragmentation and storage options.*

For more information about fragment-level authority, see the GRANT FRAGMENT and REVOKE FRAGMENT statements in the *Informix Guide to SQL: Syntax*.

SE

In INFORMIX-SE, the **-ss** option generates the pathname where the table was created if the table is not in the database directory. ♦

IUS

Using the -u Option

When you specify the **dbschema -u** option, the output displays any user-defined and complex data types contained in the database. Two suboptions, **i** and **a,** let you display the type inheritance and all the functions and casts defined over a type respectively.

The following command displays all the user-defined and complex data types for the **stork** database:

```
dbschema -d stork -u all
```

Output from **dbschema** that is executed with the specified option `-u all` might appear as shown in the following example:

```
create row type 'informix'.person_t
    (
    name varchar(30, 10) not null,
    address varchar(20, 10),
    city varchar(20, 10),
    state char(2),
    zip integer,
    bdate date
    );
create row type 'informix'.employee_t
    (
    salary integer,
    manager varchar(30, 10)
    ) under person_t;
```

The following command displays the user-defined and complex data types, as well as their type inheritance for the **person_t** table in the **stork** database:

```
dbschema -d stork -ui person_t
```

Output from **dbschema** executed with the option `-ui person_t` might appear as shown in the following example:

```
create row type 'informix'.person_t
    (
    name varchar(30, 10) not null,
    address varchar(20, 10),
    city varchar(20, 10),
    state char(2),
    zip integer,
    bdate date
    );
create row type 'informix'.employee_t
    (
    salary integer,
    manager varchar(30, 10)
    ) under person_t;
create row type 'informix'.sales_rep_t
    (
    rep_num integer,
    region_num integer,
    commission decimal(16),
    home_office boolean
    ) under employee_t;
```

The following command displays the user-defined and complex data types including all functions and class defines over a type for the **per_udt** table in the **stork** database:

```
dbschema -d stork -ua per_udt
```

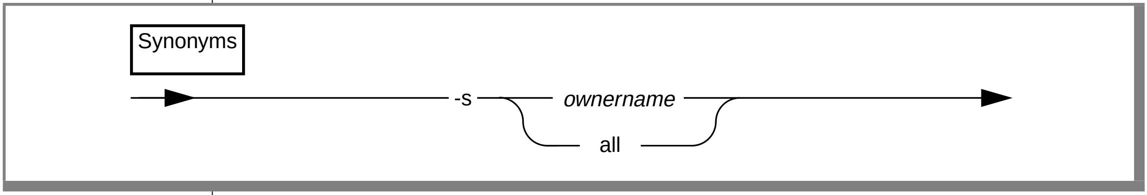
Output from **dbschema** executed with the option `-ua per_udt` might appear as shown in the following example:

```
create opaque type 'informix'.per_udt
     (
     internallength=95,
     alignment=1
     );

create implicit cast (lvarchar as per_udt with 'informix'.per_in);
CREATE FUNCTION "informix".per_in(l lvarchar)
     RETURNS per_udt
     EXTERNAL NAME '$USERFUNCDIR/person.so(Person_In)'
     LANGUAGE C ;

create implicit cast (lvarchar as per_udt with 'informix'.per_out);
CREATE FUNCTION "informix".per_out(p per_udt)
     RETURNS lvarchar
     EXTERNAL NAME '$USERFUNCDIR/person.so(PERSON_OUT)'
     LANGUAGE C NOT VARIANT;
```

Obtaining the Synonym Schema



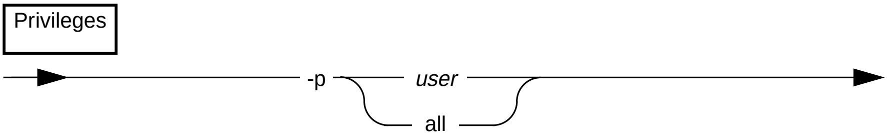
Element	Purpose	Key Considerations
-s *ownername*	Displays the CREATE SYNONYM statements owned by *ownername.*	None.
-s all	Displays all CREATE SYNONYM statements for the database, table, or view specified.	None.

Output from **dbschema** that is executed with the specified option `-s alice` might appear as shown in the following example:

```
CREATE SYNONYM 'alice'.cust FOR 'alice'.customer
```

For more information about the CREATE SYNONYM statement, see the *Informix Guide to SQL: Syntax*.

Obtaining the Privilege Schema



Element	Purpose	Key Considerations
-p *user*	Displays the GRANT statements that grant privileges to a user where *user* can be a user name or role name. Specify only one user or role.	**Restriction:** You cannot specify a specific list of users with the **-p** option. You can specify either one user or role, or all users and roles.
-p all	Displays the GRANT statements for all users for the database, table, or view specified, or to all roles for the table specified.	None.

You cannot specify a specific list of users with the **-p** option. You can specify either one user or all users. The output also displays any GRANT FRAGMENT statements that are issued for a particular user or role or the entire schema.

In the **dbschema** output, the AS keyword indicates the grantor of a GRANT statement. The following example output indicates that **norma** issued the GRANT statement:

```
GRANT ALL ON 'tom'.customer TO 'claire' AS 'norma'
```

When the GRANT and AS keywords appear in the **dbschema** output, you might need to grant privileges before you run the **dbschema** output as an SQL script. Referring to the previous example output line, the following conditions must be true before you can run the statement as part of a script:

- **norma** must have the Connect privilege to the database.
- **norma** must have all privileges WITH GRANT OPTION for the table **tom.customer**.

For more information about the GRANT, GRANT FRAGMENT, and REVOKE FRAGMENT statements, see the *Informix Guide to SQL: Syntax*.

Displaying Privilege Information for a Role

A *role* is a classification with privileges on database objects granted to the role. The DBA can assign the privileges of a related work task, such as an engineer, to a role and then grant that role to users, instead of granting the same set of privileges to every user. After a role is created, the DBA can use the GRANT statement to grant the role to users or to other roles.

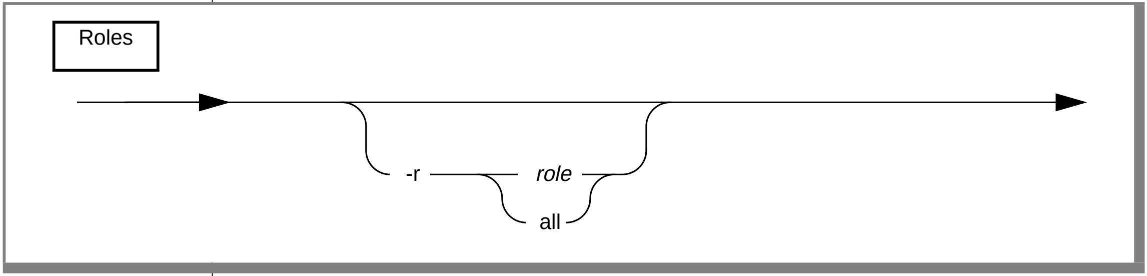
The following **dbschema** command and output show the privileges that were granted for the **calen** role:

```
sharky% dbschema -p calen -d stores7
```

```
DBSCHEMA Schema Utility       INFORMIX-SQL Version 7.22
Copyright (C) Informix Software, Inc., 1984-1996
Software Serial Number RDS#N000000

grant alter on table1 to 'calen'
```

Obtaining the Role Schema



Element	Purpose	Key Considerations
-r *role*	Displays the CREATE ROLE and GRANT statements that are needed to replicate and grant the specified role.	**Restriction:** You cannot specify a list of users or roles with the **-r** option. You can specify either one role, or all roles. SE does not support the **-r** option.
-r all	Displays all CREATE ROLE and GRANT statements that are needed to replicate and grant all roles.	None.

Displaying Information on Roles

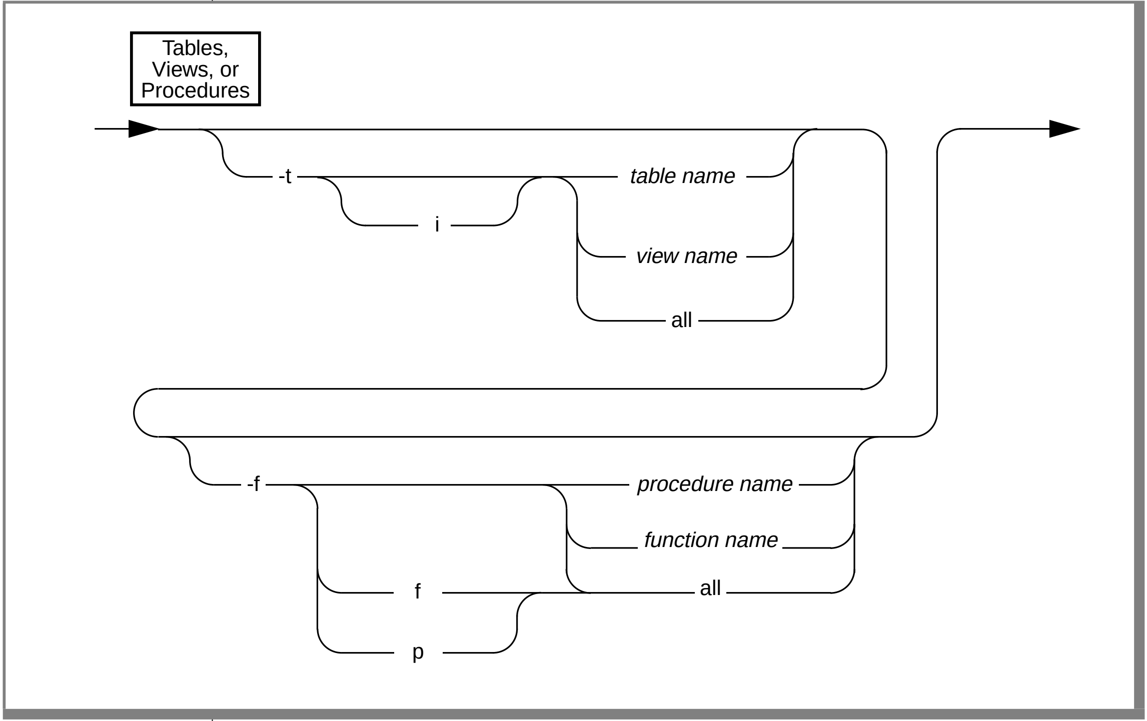
The following **dbschema** command and output show that the role **calen** was created and was granted to **cathl**, **judith,** and **sallyc**:

```
sharky% dbschema -r calen -d stores7
```

```
DBSCHEMA Schema Utility       INFORMIX-SQL Version 7.22
Copyright (C) Informix Software, Inc., 1984-1996
Software Serial Number RDS#N000000
create role calen;

grant calen to cathl with grant option;
grant calen to judith ;
grant calen to sallyc ;
```

Specifying a Table, View, or Procedure



Element	Purpose	Key Considerations
-f all	Limits the SQL statement output to those statements that are needed to replicate all functions and procedures.	None.
-f *function name*	Limits the SQL statement output to only those statements that are needed to replicate the specified function.	None.
-f *procedure name*	Limits the SQL statement output to only those statements that are needed to replicate the specified procedure.	None.
-ff all	Limits the SQL statement output to those statements that are needed to replicate all functions.	None.
-fp all	Limits the SQL statement output to those statements that are needed to replicate all procedures.	None.
-t *table name*	Limits the SQL statement output to only those statements that are needed to replicate the specified table.	None.

(1 of 2)

Element	Purpose	Key Considerations
-t *view name*	Limits the SQL statement output to only those statements that are needed to replicate the specified view.	None.
-t all	Includes in the SQL statement output all statements that are needed to replicate all tables and views.	None.
-ti *table name*	Includes in the SQL statement output all statements that are needed to replicate all table levels.	None.
-ti all	Includes in the SQL statement output all statements that are needed to replicate all tables and views. Functionally equivalent to **-t all**.	None.

(2 of 2)

For more information about the CREATE PROCEDURE and CREATE FUNCTION statements, see the *Informix Guide to SQL: Syntax*.

Using the -ss Option to Obtain Table Information

When you use the **-ss** option, you can retrieve information about fragmented tables, the lock mode, and extent sizes.

The following **dbschema** output shows the expressions specified for fragmented table.

```
DBSCHEMA Schema Utility       INFORMIX-SQL Version 7.20.UC1
Copyright (C) Informix Software, Inc., 1984-1995
{ TABLE “sallyc”.t1 row size = 8 number of columns = 1 index size = 0 }
create table “sallyc”.t1
(
c1 integer
) fragment by expression
(c1 < 100 ) in db1 ,
((c1 >= 100 ) AND (c1 < 200 ) ) in db2 ,
remainder in db4
extent size 16 next size 16 lock mode page;
revoke all on “sallyc”.t1 from “public”;
```

Displaying the Distribution Information for Tables

To display the distribution information that is stored for a table in a database, use the **-hd** option with the name of the table. If you specify the ALL keyword for the table name, the distributions for all the tables in the database are displayed.

Distribution information is stored only if you have run the UPDATE STATISTICS...MEDIUM or HIGH statement for one or more columns of a table. For information about the UPDATE STATISTICS statement, refer to the *Informix Guide to SQL: Syntax*.

The output of **dbschema** for distributions is provided in the following parts:

- Distribution description
- Distribution information
- Overflow information

Each section of **dbschema** output is explained in the following sections. As an example, the discussion uses the following distribution for the fictional table called **invoices**. This table contains 165 rows, including duplicates.

You can generate the output for this discussion with a call to **dbschema** that is similar to the following example:

```
dbschema -hd invoices -d pubs_stores7
```

Example Output

```
DBSCHEMA Schema Utility       INFORMIX-SQL Version 7.20.UC1
Copyright (C) Informix Software, Inc., 1984-1995
{

Distribution for cathl.invoices.invoice_num

Constructed on 03/10/1995

High Mode, 10.000000 Resolution


--- DISTRIBUTION ---

     (                        5)
  1: (  16,     7,           11)
  2: (  16,     6,           17)
  3: (  16,     8,           25)
  4: (  16,     8,           38)
  5: (  16,     7,           52)
  6: (  16,     8,           73)
  7: (  16,    12,           95)
  8: (  16,    12,          139)
  9: (  16,    11,          182)
 10: (  10,     5,          200)


--- OVERFLOW ---

  1: (   5,                56)
  2: (   6,                63)
}
```

Distribution Description

The first part of the **dbschema** output describes which data distributions have been created for the specified table. The name of the table is stated in the following example:

```
Distribution for cathl.invoices.invoice_num
```

The output is for the **invoices** table, which is owned by the user `cathl`. This data distribution describes the column **invoice_num**. If a table has distributions that are built on more than one column, **dbschema** lists the distributions for each column separately.

The date on which the distributions are constructed is listed. In this example, the date is 03/10/1995, which is the date when the UPDATE STATISTICS statement that generated the distributions was executed. You can use this date to tell how outdated your distributions are. Although the system records the date, it does not record the time.

The last line of the description portion of the output describes the mode (medium or high) in which the distributions were created, and the resolution. If you create the distributions with medium mode, the confidence of the sample is also listed. For example, if the UPDATE STATISTICS statement is executed with high mode with a resolution of 10, the last line appears as shown in the following example:

```
High Mode, 10.000000 Resolution
```

Distribution Information

The distribution information describes the bins that are created for the distribution, the range of values in the table and in each bin, and the number of distinct values in each bin. Consider the following example:

```
       (                           5)
    1: (   16,      7,            11)
    2: (   16,      6,            17)
    3: (   16,      8,            25)
    4: (   16,      8,            38)
    5: (   16,      7,            52)
    6: (   16,      8,            73)
    7: (   16,     12,            95)
    8: (   16,     12,           139)
    9: (   16,     11,           182)
   10: (   10,      5,           200)
```

The first value in the rightmost column is the smallest value in this column. In this example, it is 5.

The column on the left shows the bin number, in this case 1 through 10. The first number in the parentheses shows how many values are in the bin. For this table, 10 percent of the total number of rows (165), is rounded down to 16. The first number is the same for all the bins except for the last. The last row might have a smaller value, indicating that it does not have as many row values. In this example, all the bins contain 16 rows except the last one, which contains 10.

The middle column within the parentheses indicates how many distinct values are contained in this bin. Thus, if there are 11 distinct values for a 16-value bin, it implies that one or more of those values are duplicated at least once.

The right column within the parentheses is the highest value in the bin. The highest value in the last bin is also the highest value in the table. For this example, the highest value in the last bin is 200.

Overflow Information

The last portion of the **dbschema** output shows values that have many duplicates. The number of duplicates of indicated values must be greater than a critical amount that is determined as approximately 25 percent of the resolution times the number of rows. If left in the general distribution data, the duplicates would skew the distribution, so they are moved from the distribution to a separate list, as shown in the following example:

```
--- OVERFLOW ---

    1: (    5,           56)
    2: (    6,           63)
```

For this example, the critical amount is 0.25 * 0.10 * 165, or 4.125. Therefore, any value that is duplicated five or more times is listed in the overflow section. Two values in this distribution are duplicated five or more times in the table: the value 56 is duplicated five times, and the value 63 is duplicated six times.

Using DB-Access with dbschema Output

You can use the **dbschema** utility to get the schema of a database and redirect the **dbschema** output to a file. Later, you can feed this file to DB-Access to re-create the database.

Inserting a Table into a Database Example

The following example copies the CREATE TABLE statements for the customer table into the **dbschema** output file, **tab.sql**.

```
dbschema -d db -t customer > tab.sql
```

Remove the header information about **dbschema** from the output file, **db.sql**. Then use DB-Access to re-create the table in another database:

```
dbaccess db1 tab.sql
```

Recreating the Schema in Another Database

The following example puts the statements for creating the entire database into the **dbschema** output file.

1. Remove the header information about **dbschema** from the output file, in this case, db.sql.
2. Add a CREATE DATABASE statement at the beginning of the output file or use DB-Access to create a new database.
3. Use DB-Access to re-create the schema in a new database.

   ```
   dbschema -d db > db.sql
   dbaccess testdb db.sql
   ```
4. You can also use the **-ss** option.

   ```
   dbscema -d db -ss > db.sql
   ```
5. Edit the file db.sql and remove the **dbschema** header information.
6. Use DB-Access to re-create the schema in a new database.

   ```
   dbschema testdb db.sql
   ```

   When you use db.sql on another database server, confirm that dbspaces exist.

Now you have databases **db** and **testdb**, which differ in name but have the same schema.

Object Modes and Violation Detection

The **dbschema** output supports object modes and violation detection:

- The output shows the names of not-null constraints after the not-null specifications. You can use the output of the utility as input to create another database. If the same names were not used for not-null constraints in both databases, problems could result.
- The output shows the object mode of objects that are in the disabled state. These objects can be constraints, triggers, or indexes.
- The output shows the object mode of objects that are in the filtering state. These objects can be constraints or unique indexes.
- The output shows the violations and diagnostics tables that are associated with a base table (if violations and diagnostics tables were started for the base table).

For more information about object modes and violation detection, see the SET, START VIOLATIONS TABLE, and STOP VIOLATIONS TABLE statements in the *Informix Guide to SQL: Syntax*.

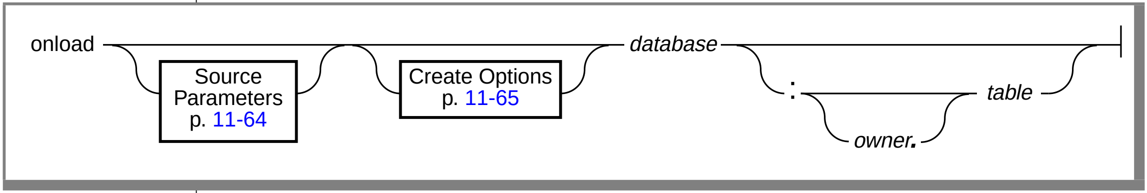
The onload Utility

The **onload** utility creates a database or table in a specified dbspace (excluding INFORMIX-SE). Then **onload** loads it with data from an input tape or disk file that the **onunload** utility creates.

AD/XP
XPS

To load and unload data in Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options or OnLine XPS, use external tables formatted in the Informix internal data representation format. You can load and unload files with the default delimiter (|) format. For complete information on external tables, see the *Administrator's Guide for Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options* or the *INFORMIX-OnLine XPS Administrator's Guide*. For syntax, see the *Informix Guide to SQL: Syntax*. ♦

During the load, you can move simple large objects that are stored in a blobspace to another blobspace.

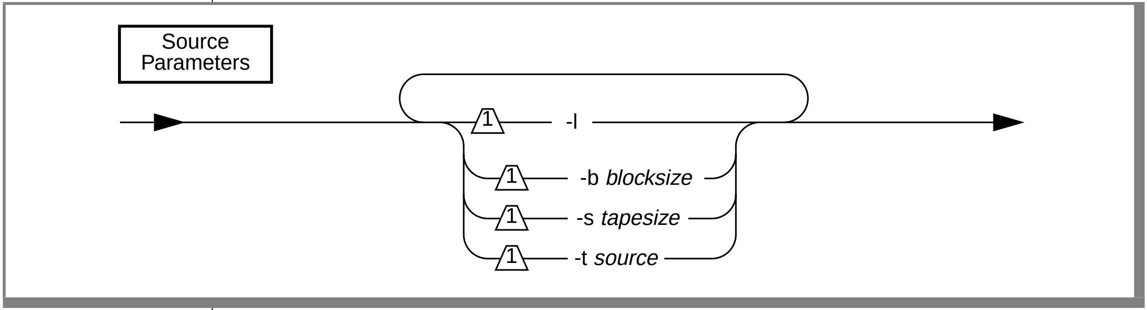


Element	Purpose	Key Considerations
database	Specifies the name of the database.	**Restriction**: The database name cannot include a database server name (*database@dbservername*). **References**: Syntax must conform to the Identifier segment; see the *Informix Guide to SQL: Syntax*.
owner.	Specifies the owner of the table.	**Restriction**: The owner name must not include illegal characters. **References**: For pathname syntax, refer to your operating-system documentation.
table	Specifies the name of the table.	**Restriction**: The table must not exist. **References**: Syntax must conform to the Table Name segment; see the *Informix Guide to SQL: Syntax*.

If you do not specify any source-parameter options, **onload** uses the device that is specified as TAPEDEV. The block size and tape size are the values that are specified as TAPEBLK and TAPESIZE, respectively. (For more information about TAPEDEV, TAPEBLK, and TAPESIZE, refer to your *Administrator's Guide*.)

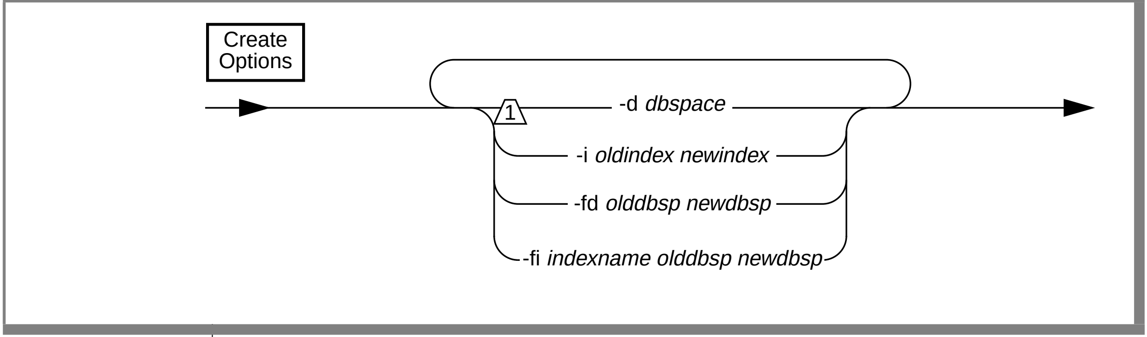
If you do not specify creation options, **onload** stores the database or table in the root dbspace.

Specify Source Parameters



Element	Purpose	Key Considerations
-b *blocksize*	Specifies in kilobytes the block size of the tape device.	**Restrictions**: Unsigned integer. Must specify the block size of the tape device. **Additional Information**: This option overrides the default value in TAPEBLK or LTAPEBLK.
-l	Directs **onload** to read the values for tape device, block size, and tape size from the configuration parameters LTAPEDEV, LTAPEBLK, and LTAPESIZE, respectively.	**Additional Information**: If you specify **-l**, and then **-b**, **-s**, or **-t**, the value that you specify overrides the value in the configuration file.
-s *tapesize*	Specifies in kilobytes the amount of data that the database server can store on the tape.	**Restrictions**: Unsigned integer. Must specify the amount of data that the database server can store on the tape. **Additional Information**: This option overrides the default value in TAPESIZE or LTAPESIZE.
-t *source*	Specifies the pathname of the file on disk or of the tape device where the input tape is mounted.	**Restriction**: Must be a legal pathname. **Additional Information**: This option overrides the tape device that TAPEDEV or LTAPEDEV specifies. **References**: For pathname syntax, see your operating-system documentation.

Create Options



Element	Purpose	Key Considerations
-d *dbspace*	Specifies the dbspace where the database or table will be stored.	**Restriction**: The dbspace must exist.
-fd *olddbspace newdbspace*	Allows you to move a data fragment from one dbspace to another.	**Restriction**: The new dbspace must exist and not already contain another data fragment for the table. **Additional Information**: This option is used with parallel data query (PDQ) and table fragmentation.
-fi *indexname olddbspace newdbspace*	Allows you to move index fragments from one dbspace to another.	**Restriction**: The new dbspace must exist and not already contain another index fragment for the table. **Additional Information**: This option is used with PDQ and table fragmentation.
-i *oldindex newindex*	Directs **onload** to rename the table index when it stores the index on disk.	**Restriction**: You must specify a table name in the command line. **Additional Information**: Use the **-i** option to rename indexes during the load to avoid conflict with existing index names. **References**: Syntax must conform to the Identifier segment; see the *Informix Guide to SQL: Syntax*.

If you do not specify any create options, the **onload** utility stores the database or table in the root dbspace.

You can use the **-d**, **-i**, **-fd**, and **-fi** options in any order and as often as necessary as long as you use unique pairs.

Constraints That Affect onload and onunload

You can use **onunload** and **onload** to move data between databases if the NLS and GLS locales are identical. For example, if user A has a French locale NLS table on server A, and tries to load data into a German locale GLS table on server B, **onload** and **onunload** reports errors. However, if both the NLS and GLS tables were created with the same French locale, **onload** and **onunload** would work.

The tape that **onload** reads contains binary data that is stored in disk-page-sized units. For this reason, the computers where the original database resides (where you use **onunload**) and where the target database will reside (where you use **onload**) must have the following characteristics:

- The same page size
- The same representation of numeric data
- The same byte alignment for structures and unions

If the page sizes are different, **onload** fails. If the alignment or numeric data types on the two computers are different (for example, with the most-significant byte last instead of first, or different float-type representations), the contents of the data page could be misinterpreted.

Restrictions That Affect onload and onunload

The **onload** and **onunload** utilities have the following restrictions:

- The original database and the target database must be from the same version of the database server.
- You cannot use **onload** and **onunload** to move data between non-GLS and GLS locales.

IUS

UD

- Do not use **onload** and **onunload** to move data between two Universal Server databases if they contain extended data types. Use the HPL instead to move the Informix Dynamic Server with Universal Data Option or Universal Server data. However, you can use **onload** and **onunload** with this data if the databases contain legacy data types. ♦

SE

- INFORMIX-SE does not support **onload** and **onunload**. ♦

Important: *You cannot use the **onload** and **onunload** utilities to move data from one version to another. You also cannot use these utilities to move data between different types of servers.*

Constraints That Affect onload

The **onload** utility performs faster than the **dbimport**, **dbload**, or LOAD methods. In exchange for this higher performance, **onload** has the following constraints:

- The **onload** utility can only create a new database or table; you must drop or rename an existing database or table of the same name before you run **onload**. The **onload** utility prompts you to rename blobspaces during execution, if desired.
- The **onload** utility places a shared lock on each of the tables in the database during the load. While you cannot update a table row with the lock in place, the database is available for queries.
- When you load a complete database, the user who executes **onload** becomes the owner of the database.
- The **onload** utility creates a database without logging; you must initiate logging after **onload** loads the database.
- When you use **onload** to load a table into a logged database, you must turn off logging for the database during the operation.

Logging While Using onload

The **onload** utility performs all of its loading within a transaction. This feature allows the changes to be rolled back if an error occurs.

When you use **onload** to create tables from an **onunload** input tape, **onload** can only load information into a database without logging. Thus, before you load a table into an existing, logged database, end logging for the database. You also might want to consider loading during off-peak hours. Otherwise, you might fill the logical-log files or consume excessive shared-memory resources. After you load the table, create a level-0 dbspace backup before you resume database logging.

When you use **onload** to create databases from an **onunload** input tape, the databases that result are not ANSI compliant and do not use transaction logging. You can make a database ANSI compliant, or add logging, after you load the database. (For more information about logging, refer to the *Informix Guide to SQL: Reference*.)

Moving Simple Large Objects or BYTE and TEXT Data to a Blobspace

If you load a table that contains simple large objects or BYTE and TEXT data stored in a blobspace, **onload** asks you if you want to move them to another blobspace. If you respond `yes`, **onload** displays the blobspace name where the simple large objects or BYTE and TEXT data were stored when the tape was created. It then asks you to enter the name of the blobspace where you want the simple large objects or BYTE and TEXT data stored. If you enter a valid blobspace name, **onload** moves all simple large object (BYTE and TEXT) columns in the table to the new blobspace. Otherwise, **onload** prompts you again for a valid blobspace name.

Ownership and Privileges

When you load a new database, the user who runs **onload** becomes the owner. Ownership within the database (tables, views, and indexes) remains the same as when the database was unloaded to tape with **onunload**.

To load a table, you must have the Resource privilege on the database. When **onload** loads a new table, the user who runs **onload** becomes the owner unless you specify an owner in the table name. (You need the DBA privilege for the database to specify an owner in the table name.)

The **onunload** utility does not preserve synonyms or access privileges. To obtain a listing of defined synonyms or access privileges, use the **dbschema** utility, described on page 11-44, before you run **onunload**.

Exclusive Locking During Load Operation

During the load operation, **onload** places an exclusive lock on the new database or table. Loading proceeds as a single transaction, and **onload** drops the new database or table if an error or system failure occurs.

The onmode Utility

This section discusses **onmode -b.** For information about the other **onmode** options, refer to your *Administrator's Guide.*

When you upgrade a database server, several modifications make the format of the database incompatible with the older version. The **onmode -b** command restores the databases to a format that is compatible with the earlier version. You must revert the databases before users can access the data with the earlier database server version. The utility does not revert changes made to the layout of the data that do not affect compatibility.

UNIX

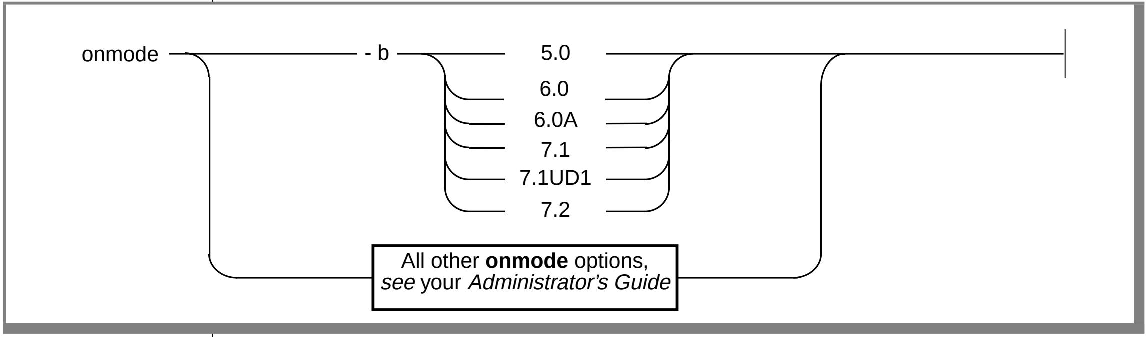
You must be user **root** or user **informix** to execute **onmode**. ♦

WIN NT

You must be a member of the **Informix-Admin** group to execute **onmode**. ♦

Preparing to Use the -b Option

Before you use the **-b** option, notify users that you are going to bring the database server off-line. The reversion utility forcibly removes all users and shuts down the database server. The **-b** option includes an implicit **-yuk**. Make sure that the **INFORMIXSERVER** environment variable is set to the correct database server.



Element	Purpose	Key Considerations
-b 5.0	Change the database to the Version 5.0 format.	**Additional Information:** Refer to "Reverting to OnLine Version 5.0" on page 11-70.
-b 6.0	Change the database to the Version 6.0 format.	**Additional Information:** Refer to "Reverting to OnLine Dynamic Server Version 6.0" on page 11-71.
-b 6.0A	Change the database to the Version 6.0 ALS format.	**Additional Information:** Refer to "Reverting to Version 6.x ALS" on page 10-44.
-b 7.1	Change the database to the Version 7.10.UC1 format, which is compatible with all 7.10.UCx formats.	**Additional Information:** Refer to "Reverting to an OnLine NLS Database Server" on page 10-43 and "Reverting from Version 7.10.UDI, 7.12, and 7.14 to 7.10.UCx" on page 5-50.
-b 7.1UD1	Change the database to the Version 7.1UD1 format, which is compatible with 7.11, 7.12, 7.13, and 7.14 formats.	**Additional Information:** Refer to "Reverting to an OnLine NLS Database Server" on page 10-43.
-b 7.2	Change the database to the Version 7.2x format.	**Additional Information:** Refer to "Reverting to Version 7.2" on page 11-72 and "Reverting to OnLine Dynamic Server" on page 5-33.

***Tip:** If you type **onmode -b -**, the available options appear.*

ODS

Reverting to OnLine Version 5.0

Before you use the **-b** option, you must free any resources that you allocated beyond Version 5.0 limits. Observe the following limits:

- The number of logical logs is less than or equal to 61.
- The number of dbspaces is less than or equal to 38.
- The number of chunks is less than or equal to 58.

  In versions of OnLine Dynamic Server before Version 6.0, the maximum number of chunks depends on the length of the chunk pathnames but cannot exceed 58.
- Remove any fragmented tables.

Depending on the version of the database server from which you are reverting, the **onmode** utility performs the following actions:

- Verifies that no GLS databases exist
- Verifies that data replication is off
- Removes the second slot in the archive reserved page for data replication
- Drops the **sysmaster** database
- Rewrites leaf pages of all indexes
- Frees reserved-page extensions
- Removes the data-replication slot from the archive reserved page
- Rewrites all **partnums** on disk (systables, database tablespace, tablespace pages, blob freemap pages) in their old formats
- Rewrites dbspace page in the old format
- Writes a Version 5.0 format checkpoint record to a clean logical-log file

The reformatting does not make the data space identical to its earlier format. Some of the changes made during conversion from an earlier version to Version 7.3 do not make the space incompatible with earlier versions, and the **-b** option does not modify these changes.

ODS

Reverting to OnLine Dynamic Server Version 6.0

Before you use the **-b** option, you must free any resources that you allocated beyond Version 6.0 limits. Observe the following limits:

- The number of page-cleaner threads is less than 33.
- The number of LRU queues is less than 32.

You set the maximum number of page-cleaner threads with the CLEANER configuration parameter and the maximum number of LRU queues with the LRUS configuration parameter.

Before you run **onmode -b 6.0**, you should also run **$INFORMIXDIR/etc/smi_unld** if either of the following conditions is true:

- You configured your database server for secure auditing.
- You used ON-Archive and want to preserve the associated catalog information.

Executing **$INFORMIXDIR/etc/smi_unld** preserves any data in the **sysmaster** database permanent tables. After you run **onmode -b**, OnLine Dynamic Server informs you of the **smi_load** script, which you can use to import your data for the permanent tables back into the **sysmaster** database once you initialize Version 6.0 of OnLine Dynamic Server.

Reversion from Version 7.3 to Version 6.0 performs the following actions:

- Verifies that none of the existing tables or indexes are fragmented
- Drops the **sysmaster** database

Reverting to Version 7.2

Only Version 7.22 or later database servers support continuous data replication (Enterprise Replication). Both Version 7.22 and later database servers and Version 9.1 and later support ON-Bar. Only Informix Dynamic Server with Universal Data Option or Universal Server support complex, user-defined, and built-in data types.

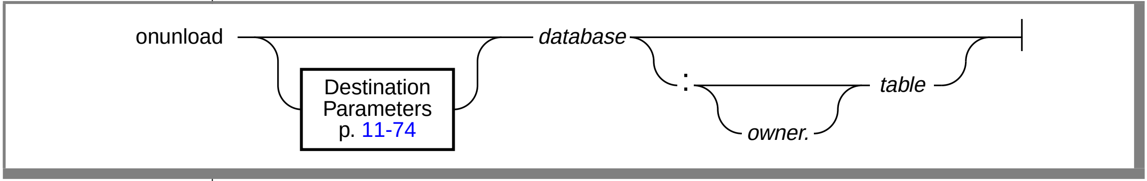
Execute **onmode -b 7.2** to revert the following database servers:

- Version 7.3, Version 7.22x, Version 7.2, or 7.21. The reversion utility automatically drops the **syscdr** database that continuous data replication uses.
- Version 9.1x to Version 7.2. The reversion utility automatically drops the new system catalog tables that Informix Dynamic Server with Universal Data Option or Universal Server created, as OnLine Dynamic Server does not use them. After the system catalog tables are reverted, you can install either OnLine Dynamic Server or OnLine Workgroup Server 7.2, 7.21, 7.22, 7.23, 7.24.

The onunload Utility

The **onunload** utility writes a database or table into a file on tape or the hard drive. The **onunload** utility unloads the data in binary form in disk-page units, making this utility more efficient than **dbexport**. You frequently use the **onunload** utility to move data between computers.

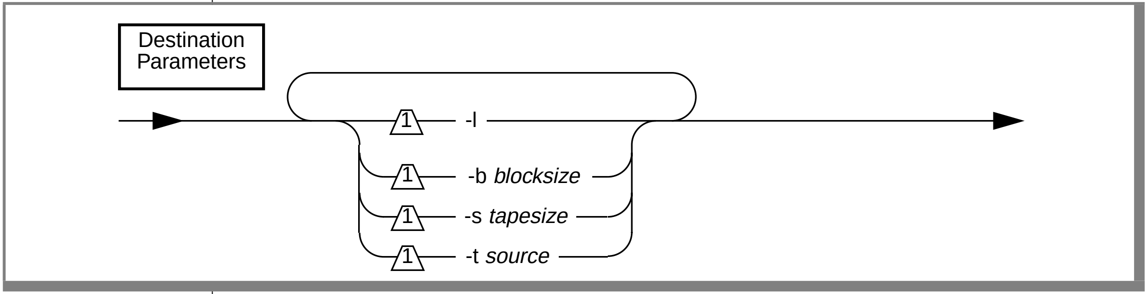
Syntax



Element	Purpose	Key Considerations
database	Specifies the name of a database.	**Additional Information**: The database name cannot include a database server name (*database@dbservername*). **References**: Syntax must conform to the Identifier segment; see the *Informix Guide to SQL: Syntax*.
owner.	Specifies the owner of the table.	**Additional Information**: The owner name must not include illegal characters. **References**: For pathname syntax, see your operating-system documentation.
table	Specifies the name of the table.	**Restriction**: The table must exist. **References**: Syntax must conform to the Table Name segment; see the *Informix Guide to SQL: Syntax*.

If you do not specify any destination parameter options, **onunload** uses the device that TAPEDEV SPECIFIES. The block size and tape size are the values specified as TAPEBLK and TAPESIZE, respectively. (For information about TAPEDEV, TAPEBLK, and TAPESIZE, refer to your *Administrator's Guide*.)

Specify Destination Parameters



Element	Purpose	Key Considerations
-b *blocksize*	Specifies in kilobytes the block size of the tape device.	**Restrictions**: The *blocksize* must be an integer. **Additional Information**: This option overrides the default value in TAPEBLK or LTAPEBLK.
-l	Directs **onunload** to read the values for tape device, block size, and tape size from LTAPEDEV, LTAPEBLK, and LTAPESIZE, respectively.	None.
-s *tapesize*	Specifies in kilobytes the amount of data that can be stored on the tape.	**Restrictions**: The *tapesize* must be an integer. **Additional Information**: This option overrides the default value in TAPESIZE or LTAPESIZE.
-t *source*	Specifies the pathname of the file on disk or of the tape device where the input tape is mounted.	**Additional Information**: This option overrides the tape device specified by TAPEDEV or LTAPEDEV. It must be a legal pathname.

Constraints That Affect onunload

The **onunload** utility can unload data more quickly than either **dbexport** or the UNLOAD statement because it copies the data in binary and in page-sized units. However, this feature places the following constraints on its use:

- You must load the data on the **onunload** tape into a database or table that your database server (excluding INFORMIX-SE) manages.
- You can use **onunload** and **onload** with Informix Dynamic Server with Universal Data Option or Universal Server if the databases contain legacy data types.

- You must load the tape that **onunload** writes onto a computer with the same page size and the same representation of numeric data as the original computer.
- You must read the file that **onunload** creates with the **onload** utility of the same version of your database server. You cannot use **onunload** and **onload** to move data from one version to another.
- When you unload a complete database, you cannot modify the ownership of database objects (such as tables, indexes, and views) until after you finish reloading the database.
- When you unload and load a table, **onunload** does not preserve access privilege, synonyms, views, constraints, triggers, or default values that were associated with the original tables. Before you run **onunload**, use the **dbschema** utility to obtain a listing of the access privilege, synonyms, views, constraints, triggers, or default values. After you finish loading the table, use **dbschema** to recreate the specific information for the table.

Unloading a Database or Table

To unload a database, you must have DBA privileges for the database or be user **informix**. To unload a table, you must either own the table, have DBA privileges for the database in which the table resides, or be user **informix**. (User **root** does not have special privileges with respect to **onunload** and **onload**.)

Unloading a Database

If you unload a database, all the tables in the database—including the system catalog tables—are unloaded. All the triggers, stored procedures, defaults, constraints, and synonyms for all the tables in the database are also unloaded.

Unloading a Table

If you unload a table, **onunload** unloads the table data and information from the following system catalog tables:

- **systables**
- **syscolumns**
- **sysindexes**
- **sysblobs**

When you unload a table, **onunload** does not unload information about constraints, triggers, or default values that are associated with a table. In addition, access privileges that are defined for the table and synonyms or views that are associated with the table are not unloaded.

Logging Mode

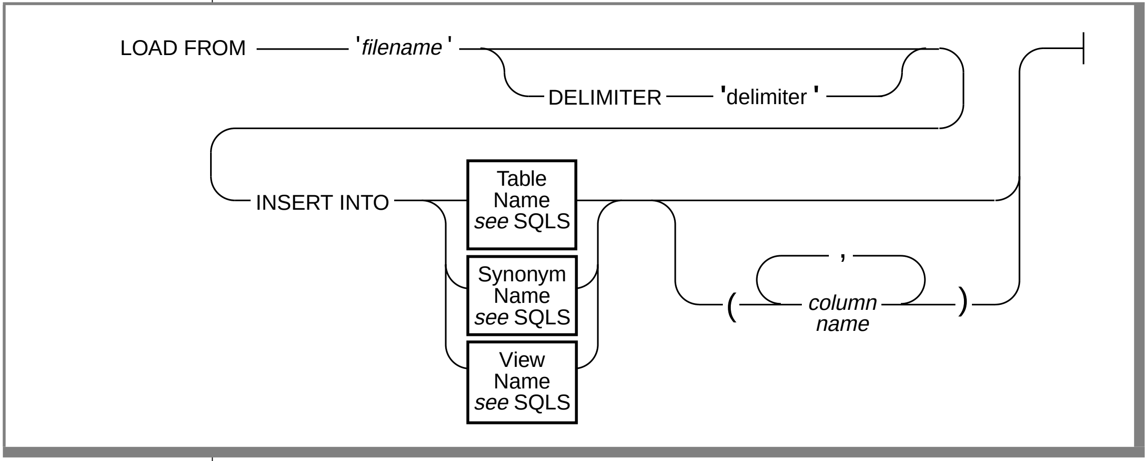
The **onunload** utility does not preserve the logging mode of a database. After you load the database with **onload**, you can make a database ANSI compliant or add logging. For information about logging modes, refer to the *Informix Guide to SQL: Syntax*.

Locking During Unload Operation

During the unload operation, the database or table is locked in shared mode. An error is returned if **onunload** cannot obtain a shared lock.

The LOAD Statement

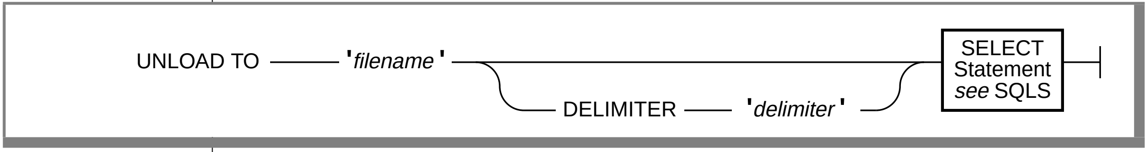
You can use the LOAD statement in DB-Access to append rows to an existing table of a database.



The preceding syntax diagram is included only for quick reference. For details about the syntax and use of the LOAD statement, refer to the *Informix Guide to SQL: Syntax*.

The UNLOAD Statement

You can use the UNLOAD statement in DB-Access to unload selected rows from a table into a text file.



The preceding syntax diagram is included only for quick reference. For details about the syntax and use of the UNLOAD statement, refer to the *Informix Guide to SQL: Syntax*.

onpload: the High-Performance Loader Utility

The **onpload** utility is the command-line portion of the High-Performance Loader (HPL). You can use the HPL with the following database servers:

- Informix Dynamic Server
- OnLine Dynamic Server, Version 7.2 or later
- Informix Dynamic Server with Universal Data Option
- Universal Server

You cannot use the HPL with SE; Informix Dynamic Server, Workgroup and Developer Editions, or OnLine Workgroup Server; or pre-7.2 versions of OnLine Dynamic Server.

For extremely large databases, the HPL has a performance advantage over other Informix data-migration utilities because it performs I/O and code set conversions in parallel. However, the user must invest significant preparation time before they use the HPL, and the HPL program itself has a significant startup time. Therefore, use the HPL only for large databases, for which the time savings in the actual loading or unloading of data makes the preparation time worthwhile. For other databases, use the data-migration tools that are discussed in this manual.

For information about how to use the HPL, refer to the *Guide to the High-Performance Loader*.

Appendix

A

Database Server Environment Variables

Various *environment variables* affect the functionality of your Informix products. You can set environment variables that identify your terminal, specify the location of your software, and define other parameters. The environment variables discussed in this chapter are listed alphabetically beginning on page A-3.

Some environment variables are required, and others are optional. For example, you must set—or accept the default setting for—certain UNIX or Windows NT environment variables.

This chapter describes how to use the environment variables that apply to Informix database servers and shows how to set them.

Types of Environment Variables

The environment variables discussed in this chapter fall into the following categories:

- Informix environment variables

  Set these standard environment variables when you want to work with Informix products. Each product manual specifies the environment variables that you must set to use that product.

UNIX

- UNIX environment variables

  Informix products rely on the correct setting of certain standard UNIX system environment variables. The **PATH** and **TERM** environment variables must always be set. You might also have to set the **TERMCAP** or **TERMINFO** environment variable to use some products effectively. ♦

WIN NT

- Windows NT environment variables

  Informix products rely on the correct setting of certain standard WINDOWS NT system environment variables. The **PATH** environment variable must always be set. ♦

GLS

- GLS environment variables

  The GLS environment variables that allow you to work in a nondefault locale are described in the *Guide to GLS Functionality*. However, these variables are also included in the list of environment variables on page A-3. ♦

NLS

- NLS environment variables (pre-7.2 products only)

  You must set some or all of these X/Open standard environment variables to benefit from NLS. These environment variables might cause your product to behave differently than when their standard Informix counterparts are set. Refer to "Native Language Support (NLS)" on page 10-10. ♦

List of Environment Variables

Figure A-1 contains an alphabetical list of the environment variables that you can set for an Informix database server. For instructions and detailed descriptions on setting these environment variables, see Chapter 4 of the *Informix Guide to SQL: Reference* for your database server. The following table uses these abbreviations for the database server names and Informix manuals:

- IDS (Informix Dynamic Server)
- IUS (INFORMIX-Universal Server and Informix Dynamic Server with Universal Data Option)
- ODS (INFORMIX-OnLine Dynamic Server)
- OWS (Informix Dynamic Server, Workgroup and Developer Editions or INFORMIX-OnLine Workgroup Server)
- SE (INFORMIX-SE)
- XPS (INFORMIX-OnLine XPS or Informix Dynamic Server with Advanced Decision Support and Extended Parallel Options)
- GLS (*Guide to GLS Functionality*)
- REF (*Informix Guide to SQL: Reference*)
- ESQL/C (*INFORMIX-ESQL/C Programmer's Manual*)

GLS

The GLS environment variables are discussed in the *Guide to GLS Functionality*. ♦

Figure A-1
Environment Variables for Informix Database Servers

Environment Variable	Restrictions	Reference
ARC_DEFAULT	IUS, ODS, IDS only (all platforms)	REF
ARC_KEYPAD	IUS, ODS, IDS only (all platforms)	REF
CLIENT_LOCALE	GLS only	GLS
CC8BITLEVEL	ESQL/C only	REF
COLLCHAR	NLS only	REF

(1 of 4)

Environment Variable	Restrictions	Reference
DBANSIWARN		REF
DBAPICODE	NLS only	REF
DBBLOBBUF	IUS, IDS, ODS, OWS only (all platforms)	REF
DBCENTURY	SQL APIS only	REF
DBDATE		REF, GLS
DBDELIMITER		REF
DBEDIT		REF
DBFLTMASK	DB-Access only	REF
DBLANG		REF, GLS
DBMONEY		REF, GLS
DBNLS	NLS only	REF
DBONPLOAD	High-Performance Loader only	REF
DBPATH		REF
DBPRINT	UNIX only	REF
DBREMOTECMD	IUS, IDS, ODS, OWS only(UNIX)	REF
DBSPACETEMP	IUS, IDS, ODS, OWS only	REF
DBTEMP	SE only	REF
DBTIME	SQL APIS only	REF
DBUPSPACE		REF
DB_LOCALE	GLS only	GLS
DELIMIDENT		REF
ENVIGNORE		REF
ESQLMF	GLS only	GLS

(2 of 4)

Environment Variable	Restrictions	Reference
FET_BUF_SIZE	SQL APIs and DB-Access only	REF
GLS8BITFSYS	SE only, GLS only	GLS
GL_DATE	GLS only	GLS
GL_DATETIME	GLS only	GLS
IFX_AUTOFREE	INFORMIX-ESQL/C Version 7.23 or later client applications	ESQL/C
INFORMIXC	ESQL/C only	REF
INFORMIXCONRETRY		REF
INFORMIXCONTIME		REF
INFORMIXCONCSMCFG	IUS only	REF
INFORMIXDIR		REF
INFORMIXKEYTAB	IUS only	REF
INFORMIXOPCACHE	OnLine/Optical only	REF
INFORMIXSERVER		REF
INFORMIXSHMBASE	IUS, IDS, ODS, OWS only(UNIX)	REF
INFORMIXSQLHOSTS		REF
INFORMIXSTACKSIZE	IUS, IDS, ODS, OWS only (all platforms)	REF
INFORMIXTERM	DB-Access only	REF
INF_ROL_SEP	IUS, ODS only	REF
LANG	NLS only	REF
LC_COLLATE	NLS only	REF
LC_CTYPE	NLS only	REF
LC_MONETARY	NLS only	REF
LC_NUMERIC	NLS only	REF

(3 of 4)

Environment Variable	Restrictions	Reference
LC_TIME	NLS only	REF
NODEFDAC		REF
ONCONFIG	IUS, IDS, ODS, OWS only (all platforms)	REF
OPTCOMPIND	IUS, IDS, ODS only (all platforms)	REF
OPTOFC	INFORMIX-ESQL/C Version 7.23 or later client applications	ESQL/C
PATH		REF
PDQPRIORITY	IUS, IDS, ODS, and XPS only (all platforms)	REF
PLCONFIG	High-Performance Loader	REF
PSORT_DBTEMP	IUS, IDS, ODS only (all platforms)	REF
PSORT_NPROCS	IUS, IDS, ODS, and XPS only (all platforms)	REF
SERVER_LOCALE	GLS only	GLS
SQLEXEC	ODS, SE only	REF
TERM	UNIX only	REF
TERMCAP	UNIX only	REF
TERMINFO	UNIX only	REF
THREADLIB	ESQL/C only	REF

(4 of 4)

Index