HP 3000 MPE/iX Computer Systems

IMAGE/SQL Administration Guide

HEWLETT PACKARD

HP Part No. 36385-90001
Printed in U.S.A. August 1997

Fifth Edition
E0897

Printing History

The following table lists the printings of this document, together with the respective release dates for each edition. The software version indicates the version of the software product at the time this document was issued. Many product releases do not require changes to the document. Therefore, do not expect a one-to-one correspondence between product releases and document editions.

Edition	**Date**	**Software Version**
First Edition	July 1990	36385-A.00.00
Second Edition	November 1992	36385-A.00.07
		or 36385-B.F0.00
Third Edition	June 1993	36385-B.F0.20
Fourth Edition	December 1994	36385B B.G0.03
Fifth Edition	August 1997	36385B B.G2.03

About this Manual

This manual explains how to use IMAGE/SQL for relational access to your TurboIMAGE/XL database. You should be familiar with TurboIMAGE/XL and have a general knowledge of relational databases to use this guide effectively. This manual assumes you are familiar with IMAGE/SQL or have read *Getting Started with HP IMAGE/SQL*.

The terms Turbo CONNECT and ALLBASE/Turbo CONNECT (ATC) are names used for previous versions of IMAGE/SQL. ATC still appears in some error messages and warnings. The former IMAGE/SQL utility, ATCUTIL, is now called IMAGESQL.

The following briefly describes each chapter:

Chapter 1 — **Introduction**

Introduces some basic IMAGE/SQL concepts and shows you how to begin using IMAGE/SQL.

Chapter 2 — **Using the IMAGE/SQL Utility**

Provides step-by-step directions for several typical administrative tasks.

Chapter 3 — **Understanding IMAGE/SQL**

Provides a discussions of IMAGE/SQL concepts and functionality.

Chapter 4 — **IMAGE/SQL Utility Commands**

Contains detailed information about the IMAGE/SQL utility commands.

Chapter 5 — **IMAGE/SQL Locking**

Describes how IMAGE/SQL assigns locks on IMAGE/SQL tables and how IMAGE/SQL handles deadlocks.

Chapter 6 — **IMAGE/SQL Transactions**

Describes IMAGE/SQL transactions, repeatable reads, and IMAGE/SQL aborted transactions.

Appendix A — **IMAGE/SQL Error Messages**

Contains reference information about error messages.

Appendix B — **SALES Database Schema**

Contains a listing of the SALES database used throughout the manual.

Appendix C — **IMAGE/SQL and Database Utilities**

Describes the DBUtil, SQLUtil, and SQLGEN utilities.

Appendix D	**SQL Exceptions**
	Lists SQL statements that have restrictions when used on a TurboIMAGE/XL data set.
Appendix E	**SQL Views for Indices**
	Describes the new views for TurboIMAGE/XL and third-party indices.
Appendix F	**DATE/TIME API**
	Describes the new externally callable procedures used to convert data to and from the internal format of ALLBASE/SQL date/time format. These new procedures are called the Date/Time Application Programming Interface or API.
Glossary	**Glossary**
	Gives basic definitions of terms.

What's New in this Edition

This edition of the *HP IMAGE/SQL Administration Guide* contains the following changes:

- Enhancements added in releases B.G1 through B.G2.03 are documented.
- Edits based on suggestions by customers and HP Support Engineers are included throughout the manual.
- Changed security for users to allow CLASS as well as PASS.
- Information on migrating a DBEnvironment previously in Appendix E is removed. The information on using SQLMigrate and SQLINSTL is in chapter 1 of the *ALLBASE/SQL Database Administration Guide* and appendix A of the *MPE/iX HP 3000 System Software Maintenance Manual*.
- A new appendix E is added on the new views for TurboIMAGE/XL and third-party indices.
- A new appendix F is added on the date/time Application Programming Interface (API).
- Other enhancements that are documented in this edition are predicate level locking, packed decimals, B-Tree indices, and the multi-connect feature.

Migrating a DBEnvironment

If you are migrating from ALLBASE/SQL version F.*nn* to version G.*nn*, you must migrate the ALLBASE/SQL DBEnvironment. An ALLBASE/SQL utility called **SQLMigrate** (SQLMIG.PUB.SYS) lets you migrate between major releases of ALLBASE/SQL. **SQLINSTL** (SQLINSTL.PUB.SYS) is a job that assists in the migration between minor releases (for example, from G.1 to G.2). For step-by-step instructions to follow in migrating a DBEnvironment using SQLMigrate or SQLINSTL, refer to chapter 1 of the *ALLBASE/SQL Database Administration Guide* or appendix A of the *MPE/iX HP 3000 System Software Maintenance Manual*.

Additional Documentation

Manuals Bundled with IMAGE/SQL	**Customer Order Number**
HP IMAGE/SQL Administration Guide	36385-90001
Getting Started with HP IMAGE/SQL	36385-90008
TurboIMAGE/XL Database Management System Reference Manual	30391-90001
ALLBASE/SQL Reference Manual	36216-90001
ALLBASE/SQL Database Administration Guide	36216-90005
ALLBASE/SQL Message Manual	36216-90009
ALLBASE/SQL Performance and Monitoring Guidelines	36216-90102
ISQL Reference Manual for ALLBASE/SQL and IMAGE/SQL	36216-90096
ODBCLINK/SE Reference Manual	36217-90406

Additional ALLBASE Manuals	**Customer Order Number**
ALLBASE/SQL Advanced Application Programming Guide	36216-90100
ALLBASE/SQL C Application Programming Guide	36216-90023
ALLBASE/SQL FORTRAN Application Programming Guide	36216-90030
ALLBASE/SQL Pascal Application Programming Guide	36216-90007
Up and Running with ALLBASE/SQL	36389-90011

Conventions

`UPPERCASE`

In a syntax statement, commands and keywords are shown in uppercase characters. The characters must be entered in the order shown; however, you can enter the characters in either uppercase or lowercase. For example:

```
COMMAND
```

can be entered as any of the following:

```
command       Command       COMMAND
```

italics

In a syntax statement or an example, a word in italics represents a parameter or argument that you must replace with the actual value. In the following example, you must replace *FileName* with the name of the file:

`COMMAND` *FileName*

punctuation

In a syntax statement, punctuation characters (other than brackets, braces, vertical bars, and ellipses) must be entered exactly as shown. In the following example, the parentheses and colon must be entered:

`(`*FileName*`):(`*FileName*`)`

<u>underlining</u>

Within an example that contains interactive dialog, user input and user responses to prompts are indicated by underlining. In the following example, <u>yes</u> is the user's response to the prompt:

`Do you want to continue? >>` <u>`yes`</u>

`{ }`

In a syntax statement, braces enclose required elements. When several elements are stacked within braces, you must select one. In the following example, you must select either ON or OFF:

$$\texttt{COMMAND} \begin{Bmatrix} \texttt{ON} \\ \texttt{OFF} \end{Bmatrix}$$

`[ ]`

In a syntax statement, brackets enclose optional elements. In the following example, OPTION can be omitted:

`COMMAND` *FileName* `[OPTION]`

When several elements are stacked within brackets, you can select one or none of the elements. In the following example, you can select OPTION or *Parameter* or neither. The elements cannot be repeated.

$$\texttt{COMMAND}\ \textit{FileName} \begin{bmatrix} \texttt{OPTION} \\ \textit{Parameter} \end{bmatrix}$$

Conventions (continued)

`[ ... ]`	In a syntax statement, horizontal ellipses enclosed in brackets indicate that you can repeatedly select the element(s) that appear within the immediately preceding pair of brackets or braces. In the example below, you can select *Parameter* zero or more times. Each instance of *Parameter* must be preceded by a comma:
	
	`[,`*Parameter*`] [...]`
	
	In the example below, you only use the comma as a delimiter if *Parameter* is repeated; no comma is used before the first occurrence of *Parameter*:
	
	`[`*Parameter*`] [,...]`

`| ... |`	In a syntax statement, horizontal ellipses enclosed in vertical bars indicate that you can select more than one element within the immediately preceding pair of brackets or braces. However, each particular element can only be selected once. In the following example, you must select A, AB, BA, or B. The elements cannot be repeated. $\left\{\begin{array}{c}\texttt{A}\\\texttt{B}\end{array}\right\}$ `| ... |`
...	In an example, horizontal or vertical ellipses indicate where portions of an example have been omitted.
`Δ`	In a syntax statement, the space symbol `Δ` shows a required blank. In the following example, *Parameter* and *Parameter* must be separated with a blank: `(`*Parameter*`)Δ(`*Parameter*`)`
( )	The symbol ( ) indicates a key on the keyboard. For example, (RETURN) represents the carriage return key.
(CTRL)*char*	(CTRL)*char* indicates a control character. For example, (CTRL)`Y` means that you press the control key and the Y key simultaneously.

Contents

Figures

Tables

1

Introduction

This chapter introduces some basic IMAGE/SQL concepts and shows how to begin using IMAGE/SQL. For additional introductory information on IMAGE/SQL, refer to *Getting Started with HP IMAGE/SQL*.

What is IMAGE/SQL?

IMAGE/SQL provides relational access to your TurboIMAGE data using the industry-standard Structured Query Language (SQL). This access method includes full read and write capability using ANSI standard functionality. Closely tuned to the architecture of HP computers, IMAGE/SQL gives you flexibility in designing and using SQL database applications on a small or large scale.

IMAGE/SQL includes the following:

- TurboIMAGE/XL.
- The components of ALLBASE/SQL needed to provide relational access to TurboIMAGE/XL databases. You receive a restricted version of ALLBASE/SQL with IMAGE/SQL. With this restricted version, the size of your TurboIMAGE/XL databases is not restricted in any way, but you can only store 12 MegaBytes (3000 4K pages) of data in user-defined SQL tables. For an unrestricted copy of ALLBASE/SQL, you must purchase ALLBASE/SQL.
- IMAGESQL, the IMAGE/SQL utility, is a data administration tool that links the TurboIMAGE/XL database to ALLBASE/SQL. To access the IMAGE/SQL utility program use the command:

  ```
  :RUN IMAGESQL.PUB.SYS
  ```

- ODBCLink/SE, the 16-bit and 32-bit ODBC compliant driver, for a client/server environment. Your client can run under Microsoft Windows 3.1 or 3.11, Windows95, or WindowsNT V3.51 or V4.0. Connection via Winsock is available in the 16-bit or 32-bit version. The driver can be used in two ways: either by direct calls from a supported Windows program or through an ODBC-compliant application (such as Microsoft Access or Visual Basic).

Basic Terms for Getting Started

The following definitions will get you started. A complete alphabetical listing of all terms and their definitions can be found in the glossary at the end of this guide.

- **ATC**—the terms Turbo CONNECT and ALLBASE/Turbo CONNECT (ATC) were used for earlier versions of IMAGE/SQL. The acronym, ATC, still appears in error messages for IMAGE/SQL (for example, ATCERR or ATCWARN).
- **ATCUtil**—is the previous name for the IMAGE/SQL utility program, now called **IMAGESQL**.
- **ISQL**—stands for Interactive Structured Query Language. With ISQL, you can interactively define and access data in an ALLBASE/SQL relational database environment. Most ALLBASE/SQL statements can be submitted interactively through ISQL, although a few statements specifically support programmatic database access and can only be submitted in application programs.
- **SQLUtil**—is the ALLBASE/SQL utility program. It is a different utility from the IMAGE/SQL utility.
- **DBEnvironment**— a collection of related files containing one or more ALLBASE/SQL databases that share the same logging and recovery process.
- **Table**—the basic unit of storage in an ALLBASE/SQL database. Tables are made up of rows and columns of data.
- **Attached Database**—a TurboIMAGE/XL database whose data can be accessed using relational access. Information about the attached TurboIMAGE/XL database is stored in the DBEnvironment.
- **Detached Database**—a TurboIMAGE/XL database whose data cannot be accessed using relational access. No information about the TurboIMAGE/XL database is stored in the DBEnvironment. A TurboIMAGE/XL database must be detached from a DBEnvironment before it is restructured, with the exception of capacity change.
- **Mapping**—the process IMAGE/SQL uses to allow a TurboIMAGE/XL database to emulate a DBEnvironment database. Mapping takes place for TurboIMAGE/XL names, data sets, data items, indices, data item types, and data security.
- **Mapped Table**—a table defined in the DBEnvironment based on a TurboIMAGE/XL data set. Data set characteristics are mapped by the IMAGE/SQL utility to ALLBASE/SQL characteristics.
- **DBC (Database Creator)**—the creator of the TurboIMAGE/XL database. You must be either the database creator or give the TurboIMAGE/XL database maintenance word to attach the database to a DBEnvironment. Commands that add users, or view or modify user information can only be executed by the DBC.

- **DBECreator**—the individual who originally configured the DBEnvironment.
- **DBA (Database Administrator)**—a database administrator of the DBEnvironment. You must be a DBA of the DBEnvironment to which the TurboIMAGE/XL database is attached to issue most IMAGE/SQL utility commands. The creator of the DBEnvironment is automatically a DBA. Other ALLBASE/SQL users can be granted DBA authority by a DBA.
- **DBEConFile**—DBEnvironment Configuration File, or DBEConFile, contains basic information that is used every time the DBEnvironment is opened. When you create a DBEnvironment, IMAGE/SQL creates the DBEConFile with the same name as the DBEnvironment itself. The DBEConFile points to the other DBEnvironment components.

Getting Acquainted with the IMAGE/SQL Utility

The following example shows how to use the IMAGE/SQL utility to attach a TurboIMAGE/XL database to a DBEnvironment.

Invoking the IMAGE/SQL Utility

To initiate an IMAGE/SQL utility session, log on to a group and account containing a TurboIMAGE/XL database and a DBEnvironment. Then at the MPE/iX system prompt, type RUN IMAGESQL.PUB.SYS. For example:

```
:HELLO NANCY.ATC
:RUN IMAGESQL.PUB.SYS

HP36385B B.G0.03    IMAGE/SQL Utility    FRI, DEC 16, 1994, 11:30 AM
(C) COPYRIGHT HEWLETT-PACKARD COMPANY 1993

>>
```

The DISPLAY OPTIONS command displays information about your IMAGE/SQL utility session:

```
>>DISPLAY OPTIONS
Current base        :
Current SQLDBE      :
Echo                : ON
Command Logging     : ON
Current Log File    : ATCLOG.SERED.ATC

>>
```

Notice that the headings "Current base:" and "Current SQLDBE:" have no information displayed at this time.

Attaching to a DBEnvironment

Three IMAGE/SQL commands are needed to attach a TurboIMAGE/XL database to the DBEnvironment:

SET TURBODB	identifies the TurboIMAGE/XL database to be attached to a DBEnvironment. To issue this command you must be the DBC or supply the TurboIMAGE/XL database maintenance word.
SET SQLDBE	identifies the DBEnvironment to which the TurboIMAGE/XL database will be attached. To issue this command you must be the DBECreator or supply the DBEnvironment maintenance word.
ATTACH	performs the attach. To issue this command you must be a DBA of the DBEnvironment.

In the following example, the SALES database and the PartsDBE DBEnvironment are identified with two SET commands.

```
>SET TURBODB SALES.SERED.ATC
>>SET SQLDBE PARTSDBE.SERED.ATC
>>
```

After the SET commands are issued, the DISPLAY OPTIONS command displays this information:

```
>>DISPLAY OPTIONS
Current base         : SALES.SERED.ATC
Current SQLDBE       : PARTSDBE.SERED.ATC
Echo                 : ON
Command Logging      : ON
Current Log File     : ATCLOG.SERED.ATC

***Database is not attached.
>>
```

The ATTACH command can now be used to attach SALES to PartsDBE. Note that messages issued at attach time inform you if mapping is taking place.

```
>>ATTACH
Split 1 compound source field(s) (ATCWARN 32063).
Mapped 15 source table/source field name(s) (ATCWARN 32062).
Mapped 1 incompatible/imprecise source type(s) (ATCWARN 32061).
>>EXIT
```

Selecting TurboIMAGE/XL Data from Mapped Tables

The attached TurboIMAGE/XL database is now a part of the DBEnvironment and can be queried with SQL SELECT statements by the DBC. (Other users must be explicitly added by the DBC. Refer to Chapter 2, "Using the IMAGE/SQL Utility," for more details.)

In the example below, the ALLBASE/SQL utility, ISQL, is used to connect to PartsDBE. A SELECT statement is then used to display the data in the mapped table SALES.VENDOR.

```
:RUN ISQL.PUB.SYS
isql=>CONNECT to 'PartsDBE';
isql=>SELECT * from sales.vendor;

select * from sales.vendor;
----------------+--------------------+------------+-----+-----
VENDOR          |STREET              |CITY        |STATE|ZIP
----------------+--------------------+------------+-----+-----
Celtic Graphics |105 19th Ave.       |Seattle     |WA   |98115
Trident 3D      |55 Homestead Road   |Cupertino   |CA   |95014
Ablrn Tech.     |90 Marina Way       |Berkeley    |CA   |94708
Space Ent.      |110 Homestead Ave.  |Cupertino   |CA   |95014
Cutler Micro    |9442 E. 57th Ave.   |Seattle     |WA   |98115
Seminational Co.|5000 Marina Way     |San Diego   |CA   |92093
.                .                    .            .     .
.                .                    .            .     .
.                .                    .            .     .
```

The multi-connect feature enables you to simultaneously connect to more than one DBEnvironment or connect to the same DBEnvironment more than once. This chapter has shown you how easy it is to get started using IMAGE/SQL. Once the database is attached, you may need to complete several additional IMAGE/SQL utility tasks. Chapter 2, "Using the IMAGE/SQL Utility," describes typical IMAGE/SQL utility tasks and shows you how to perform them.

2

Using the IMAGE/SQL Utility

This chapter assumes you have already read Chapter 1, "Introduction." This chapter provides step-by-step directions for performing IMAGE/SQL administrative tasks using IMAGE/SQL utility commands.

The following information is in this chapter:

- An overview of the role of the IMAGE/SQL administrator.
- Checklists showing which tasks to perform when setting up and maintaining the IMAGE/SQL environment.
- An explanation of how each task description is organized.
- A table showing the prerequisites needed for performing IMAGE/SQL utility tasks.
- A summary of commands needed for each task.
- Directions for performing each task.

Administering IMAGE/SQL: An Overview

As an administrator, you are responsible for the IMAGE/SQL environment. This chapter shows how to use IMAGE/SQL commands to perform tasks necessary for setting up and maintaining this environment.

Setting Up the IMAGE/SQL Environment

The initial IMAGE/SQL setup involves the following steps:

- Establishing the connection between the DBEnvironment and the TurboIMAGE/XL database.
- Adding users.
- Customizing the default mapping to meet your needs.

Maintaining the IMAGE/SQL Environment

Once the IMAGE/SQL environment is set up, nothing else is necessary until one of the following occurs:

- The needs of the IMAGE/SQL environment change.
- The TurboIMAGE/XL database requires restructuring.
- The log file becomes full.

For your convenience, Table 2-1, Table 2-2, and Table 2-3 on the next page are checklists for you to use as you set up and maintain IMAGE/SQL. Each checklist contains an ordered list of tasks with space to check off each task as it is completed. The tables indicate whether each task is optional (Opt) or required (Req). You may want to make a photocopy of these tables so you can refer to them as you perform the tasks.

Table 2-1. Checklist for Setting Up the IMAGE/SQL Environment

√	Opt/ Req	Task#	Task Description	Notes
	Opt	1	Configuring a DBEnvironment.	Perform to create a DBE—if the DBE created by IMAGESQL using the defaults is not suitable for relational access.
	Req	2	Attaching a TurboIMAGE/XL database.	Perform to establish the connection between the DBE and the TurboIMAGE/XL database for relational access.
	Opt	3	Adding users.	Perform to add users in addition to the DBC.
	Opt	4	Updating data type mapping.	Perform to select alternative data type mapping.
	Opt	5	Splitting mapped columns.	Perform to divide a mapped column into two or more mapped columns so that you can use the split column names in SQL statements.

Table 2-2. Checklist for Maintaining the IMAGE/SQL Environment

√	Opt/ Req	Task#	Task Description	Notes
	Opt	4	Updating data type mapping.	Perform to select alternative data type mapping.
	Opt	5	Splitting mapped columns.	Perform to divide a mapped column into two or more mapped columns so that you can use the split column names in SQL statements.
	Opt	6	Updating information about users.	Perform to update a user's class, password, or access mode.
	Opt	7	Deleting users.	Perform to delete an IMAGE/SQL user's access to the TurboIMAGE/XL database.
	Opt	8	Creating new log files.	Add log files when you need more log file space.

Table 2-3. Checklist for Restructuring in the IMAGE/SQL Environment

√	Opt/ Req	Task#	Task Description	Notes
	Req	9	Detaching a TurboIMAGE/XL database.	Perform before restructuring the TurboIMAGE/XL database, except when changing the capacity of the database.
	Req	NA	Restructuring the TurboIMAGE/XL database. (Some restructuring may necessitate first performing a DETACH from the DBEs.)	Refer to the *TurboIMAGE/XL Database Management System Reference Manual* for information about performing restructuring tasks.
	Req	2 - 5	Restoring the IMAGE/SQL environment; includes all setup tasks originally performed.	Perform to reattach the database, to add back users, and to redo any customized mapping.

About the Tasks

- Each task is divided into three parts:

Getting Ready	describes preparatory steps and gives information you need to know before performing the task.
Performing the Task	describes how to perform the task.
Task Reference	lists information useful when performing the task and provides cross-references to related tasks.

- To perform most tasks you must be attached to the DBEnvironment. Here are some general guidelines:
    - To issue the SET commands, you must be the creator or supply the maintenance word of the TurboIMAGE/XL database or DBEnvironment named in the command.
    - For most other commands you must be a DBA of the DBEnvironment. For commands involving database security, you must be the DBC. The maintenance word is not sufficient.
- Figure 2-1 shows the prerequisites needed to issue all IMAGE/SQL utility commands. For complete IMAGE/SQL utility command syntax, refer to Chapter 4, "IMAGE/SQL Utility Commands."

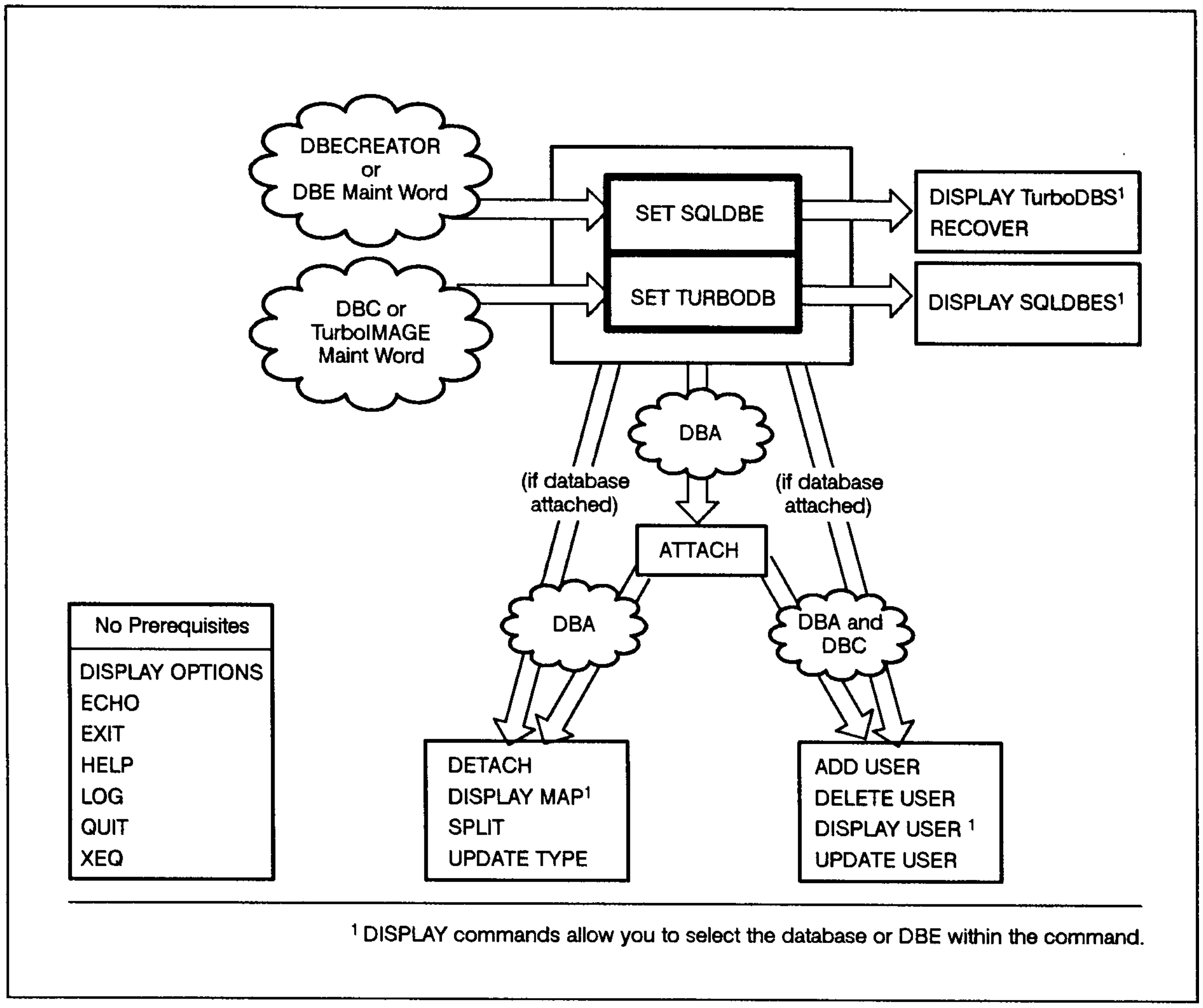


Figure 2-1. IMAGE/SQL Utility Command Prerequisites

Tasks Covered in this Chapter

For your convenience Table 2-4 lists the tasks and the commands needed to perform each task. Most of the IMAGE/SQL commands require using SET SQLDBE and/or SET TURBODB before using the commands.

Table 2-4.
IMAGE/SQL Utility Tasks and Related Commands

Task	Task #	Commands Used
Configuring a DBEnvironment	Task 1	START DBE NEW (an SQL statement)

Attaching a TurboIMAGE/XL database	Task 2	SET SQLDBE
		SET TURBODB
		ATTACH

Adding IMAGE/SQL users	Task 3	ADD USER
Updating IMAGE/SQL utility data type mapping information	Task 4	UPDATE TYPE
Splitting mapped columns	Task 5	SPLIT
Updating information about IMAGE/SQL users	Task 6	UPDATE USER
Deleting IMAGE/SQL users	Task 7	DELETE USER
Creating a log file	Task 8	ADDLOG (an SQLUtil statement)
Detaching a TurboIMAGE/XL database	Task 9	DETACH
Displaying IMAGE/SQL utility information	Task 10	DISPLAY OPTIONS DISPLAY MAP DISPLAY USER DISPLAY SQLDBE DISPLAY TURBODB HELP
Issuing MPE/iX commands from the IMAGE/SQL utility	Task 11	:*MPE/iXCommandName*
Setting IMAGE/SQL file equations	Task 12	:FILE
Logging IMAGE/SQL utility commands	Task 13	LOG
Using IMAGE/SQL utility command files	Task 14	XEQ
Selecting TurboIMAGE/XL data with SQL	Task 15	SELECT (an SQL statement)
Maintaining the ATCINFO file	Task 16	RECOVER

Task 1: Configuring a DBEnvironment

This task describes how to configure a DBEnvironment so you can access your TurboIMAGE/XL database(s) with ALLBASE/SQL. This is an optional task because the SET SQLDBE command can also create the DBEnvironment using defaults when the DBEnvironment does not exist.

Instead of using the commands as shown in the examples, you can use the IMAGE/SQL utility to create your DBEnvironment. When you run the IMAGE/SQL utility, it prompts you to create the DBEnvironment. You should still read through this section to see important information such as setting TIMEOUT values.

Getting Ready

- When all of the TurboIMAGE/XL databases to be attached to the DBEnvironment are created by the same user in one group and account, it is convenient to configure the DBEnvironment in this group and account.

  If this is not the case, several other issues should be considered. The following considerations apply if the TurboIMAGE/XL database to be attached exists in a different group and/or account than the DBEnvironment:

  - IMAGE/SQL supports standard MPE/iX security rules. Make sure correct user, group, and account capabilities are in place when you plan to access a TurboIMAGE/XL database from a DBEnvironment in a different account than the database.
  - Make sure maintenance words exist for the DBEnvironment and all TurboIMAGE/XL databases because IMAGE/SQL utility administrators often need to specify DBEnvironment and TurboIMAGE/XL maintenance words if they are not the creator.
  - Be sure to grant DBA authority to everyone who will be performing IMAGE/SQL utility tasks because IMAGE/SQL utility administrators need DBA authority to perform most IMAGE/SQL utility tasks.

- Make sure the TurboIMAGE/XL database and the DBEnvironment have the same native language support (NLS) defined for them.
- When a DBEnvironment is configured, two files are created: DBEFILE0 and DBELOG1. IMAGE/SQL requires that these files be larger than the ALLBASE/SQL defaults. Make sure these files are large enough to accommodate IMAGE/SQL, but the maximum is 5000 pages for DBEFILE0 and 10000 pages for DBELOG1. In the example below, a file size of 500 pages is used for each of these files, but you may need to adjust these sizes depending on the size and number of TurboIMAGE/XL databases you plan to attach.

Performing the Task

- Log on in the same group and account as the TurboIMAGE/XL database(s) and run ISQL to enter the statements interactively. At the ISQL prompt, enter the START DBE NEW statement. For example, to configure a DBEnvironment named PartsDBE, enter the following:

```
:RUN ISQL.PUB.SYS
isql=> START DBE 'PartsDBE' MULTI NEW
> MAXIMUM TIMEOUT = 2 MINUTES
> DEFAULT TIMEOUT = MAXIMUM
> DBEFILE0 DBEFILE DBEFILE0 <==DBEFile0Definition
> WITH PAGES = 500,         <==  .     .
> NAME = 'PartsF0',         <==  .     .
> LOG DBEFILE DBELOG1       <==DBELog0Definition
> WITH PAGES = 500,         <==  .     .
> NAME = 'PartsLog';        <==  .     .
isql=> EXIT;
:
```

  If you forget the semicolon, ISQL prompts you with a right-angle bracket (>). At this prompt, enter a semicolon and ISQL will execute the statement. If you wish to terminate the command, enter //. The MULTI parameter is necessary if you plan on a multiuser IMAGE/SQL environment.

  The above command can also be done by the IMAGE/SQL utility. The difference is that IMAGE/SQL utility automatically creates the DBEnvironment with a DBEFILE0 size of 5000 pages, log file size of 10000 pages, and other ALLBASE/SQL defaults.

- To avoid deadlock situations when there are more than two connections, you should either set the MAXIMUM TIMEOUT and DEFAULT TIMEOUT options with the START DBE NEW statement or change them with the SQLUtil ALTDBE command. The ALTDBE command updates the parameters required for DBEnvironment startup. The SQLUtil utility prompts you for necessary information. Enter a carriage return to retain an old value.

```
:RUN SQLUTIL.PUB.SYS
>>ALTDBE
DBEnvironment Name: PartsDBE
Maintenance Word:  MaintWd
:
Maximum Timeout (opt): 10  SECONDS
Default Timeout (opt): 5 SECONDS
Authorize Once per Session (opt): ON
Alter DBEnvironment Startup Parameters (y/n)? YES

DBEnvironment startup parameters altered.
>>EXIT
```

- To set a maintenance word for the newly configured DBEnvironment, use the SETDBEMAINT command of SQLUtil, as in the following example:

```
:RUN SQLUTIL.PUB.SYS
>>SETDBEMAINT
DBEnvironment Name: PartsDBE
Current Maintenance Word:  Carriage Return
New Maintenance Word:  usr
Retype New Maintenance Word:  usr
>>EXIT
```

- Once you have configured the DBEnvironment, set a maintenance word for it, and set the timeout time, you are ready to attach TurboIMAGE/XL databases to it. This is explained in Task 2.

Task Reference

- No information is given here about considerations that may be necessary when configuring a DBEnvironment that contains ALLBASE/SQL databases. Consult the *ALLBASE/SQL Database Administration Guide* for more guidance in this area.
- The following summaries of SQL statements will get you started. You can use ISQL to invoke these statements interactively.
  - □ The SQL START DBE NEW statement has the following syntax:

```
START DBE 'DBEnvironmentName' [MULTI] NEW
 ┌DUAL LOG                                                    ┐
 │BUFFER = (DataBuffPages,LogBuffPages)                       │
 │LANG = LanguageName                                         │
 │TRANSACTION = MaxTransactions                               │
 │                     ⎧ TimeoutValue ⎡SECONDS⎤ ⎫             │
 │MAXIMUM TIMEOUT =    ⎨              ⎣MINUTES⎦ ⎬             │
 │                     ⎩ NONE                   ⎭             │
 │                     ⎧ TimeoutValue ⎡SECONDS⎤ ⎫             │
 │DEFAULT TIMEOUT =    ⎨              ⎣MINUTES⎦ ⎬             │
 │                     ⎩ MAXIMUM                ⎭             │
 │DBEFile0Definition                                          │
 └DBELogDefinition                                            ┘
 |, ... |
```

    Refer to the example discussed earlier in this section for details of the syntax for *DBEFile0Definition* and *DBELogDefinition*.

  - □ When you issue the START DBE statement, a fileset named IMAGESQL is created. You can add up to 3000 pages to this fileset.

    These are the maximum pages for a DBEnvironment created by IMAGE/SQL as of version B.G2:

Log pages	10000 pages
DBEFile	5000 pages
SQL tables in fileset named IMAGESQL	3000 pages

  - □ The SQL GRANT statement has the following syntax for granting DBA authority:

```
      ⎧ CONNECT  ⎫    ⎧ DBEUserID ⎫
GRANT ⎨ DBA      ⎬ TO ⎨ GroupName ⎬ [, ... ]
      ⎩ RESOURCE ⎭    ⎩ ClassName ⎭
```

- Use SQLUtil to set a DBEnvironment maintenance word. First access SQLUtil, then use the SQLUtil SETDBEMAINT command to set a maintenance word. The syntax for this command is as follows:

```
:RUN SQLUTIL.PUB.SYS
>>SETDBEMAINT
DBEnvironment Name:  DBEnvironmentName
Current Maintenance Word:  OldMaintenanceWord
New Maintenance Word:  NewMaintenanceWord
Retype New Maintenance Word:  NewMaintenanceWord
>>EXIT
```

When no current maintenance word exists, enter a carriage return at the "Current Maintenance Word:" prompt.

Task 2: Attaching a TurboIMAGE/XL Database

This task describes how to attach a TurboIMAGE/XL database.

Note

If you are attaching a database with the same name but in a different group and/or account as a database already attached to the DBEnvironment, you must specify an alternative owner name at attach time. This is because in a mapped table, by default, the owner name is the database name. Duplicate table names are not allowed within the same database. In any case, you cannot attach the same TurboIMAGE/XL database twice to the same DBEnvironment.

Getting Ready

- Exit from SQLUtil after you finish Task 1, then run the IMAGE/SQL utility, IMAGESQL.
- It is convenient to have the TurboIMAGE/XL database and the DBEnvironment in the same group and account.

  If this is not the case, there are several issues to consider. The following considerations apply if the TurboIMAGE/XL database(s) to be attached exist(s) in a different group and/or account than the DBEnvironment:

  - IMAGE/SQL supports standard MPE/iX security rules. Correct user, group, and account capabilities must be in place to use IMAGE/SQL to access a TurboIMAGE/XL database from a DBEnvironment in a different account than the database.
  - IMAGE/SQL utility administrators often need to specify DBEnvironment and TurboIMAGE/XL maintenance words as a part of the SET command if they are not the creator. Because of this, it is recommended that maintenance words exist for the DBEnvironment and all TurboIMAGE/XL databases.
  - IMAGE/SQL utility administrators need DBA authority to perform most IMAGE/SQL utility tasks. Be sure to grant DBA authority to everyone who will be performing IMAGE/SQL utility tasks.
- The TurboIMAGE/XL database and the DBEnvironment must have the same native language support (NLS) defined for them.
- Be sure that all processes accessing the DBEnvironment are shutdown before using the ATTACH command. The first ATTACH command using the IMAGE/SQL utility requires exclusive access to the DBEnvironment.

- The ATTACH command also requires that the appropriate SET SQLDBE and SET TURBODB commands have been issued. To check the status of these commands, run the IMAGE/SQL utility and use the DISPLAY OPTIONS command:

```
:RUN IMAGESQL.PUB.SYS
>>DISPLAY OPTIONS
Current base      :
Current SQLDBE    :
Echo              : ON
Command Logging   : ON
Current Log File  : ATCLOG.SERED.ATC
```

- If necessary, issue the SET commands in IMAGE/SQL utility. For example:

```
>>SET SQLDBE PARTSDBE.SERED.ATC
>>SET TURBODB SALES.SERED.ATC
>>
```

If the DBEnvironment does not exist, IMAGE/SQL displays this message:

```
DBE does not exist, do you want to create one? [Y/N] :
```

If you reply 'Y', a DBEnvironment and DBEFiles are automatically created for you. The files created are:

File Created	File Name
DBEConFile	*DBEnvironmentName*
DBEFile	*DBEnvironmentName*`FL`
LOG file	*DBEnvironmentName*`LG`
ATCINFO file	*DBEnvironmentName*`CR`

For example, if you issue the command

```
SET SQLDBE MYDBE
```

and the DBEnvironment MYDBE does not exist, these files are created: MYDBE, MYDBEFL, MYDBELG, and MYDBECR.

The size of MYDBEFL is 5000 pages, and that of MYDBELG is 10000 pages.

Performing the Task

When you are sure the appropriate SET commands have been specified and that the correct MPE/iX security is in place, issue the ATTACH command.

```
:RUN IMAGESQL.PUB.SYS
>>ATTACH
Split 1 compound source fields (ATCWARN 32063).
Mapped 15 source table/source field names (ATCWARN 32062).
Mapped 1 incompatible source types (ATCWARN 32061).
>>
```

Messages issued at attach time inform you if any mapping has been done. The SALES database is now a logical part of the PartsDBE DBEnvironment. Although the data remains in the TurboIMAGE/XL database, it can now be accessed from mapped tables just as it would be accessed from ALLBASE/SQL tables.

Task Reference

- By default, the IMAGE/SQL utility uses the TurboIMAGE/XL database name as the owner name.

  You must specify an alternative owner name if you are attaching a TurboIMAGE/XL database with the same name as one already attached. To do this, use the WITH OWNER= parameter of the ATTACH command.
- To specify a maintenance word, use the MAINT= parameter of the SET TURBODB or SET SQLDBE command. Refer to Chapter 4, "IMAGE/SQL Utility Commands," for details about these commands and their parameters.
- Use the DISPLAY MAP command to see detailed database mapping information.
- When a TurboIMAGE/XL database is attached to a DBEnvironment, IMAGE/SQL performs the following tasks:
  - Makes a table entry in the system catalog of the DBEnvironment for each corresponding source data set.
  - Creates a column definition for each field in the source data set.

    IMAGE/SQL columns are defined as NOT NULL with default values.

    Default values are based on IMAGE/SQL data types, as listed in Table 2-5.

Table 2-5. IMAGE/SQL Default Data Types

Group	IMAGE/SQL Data Type	Default Type
Alphanumeric	CHAR	Blanks
Date/Time	DATE	CURRENT_DATE
	DATETIME	CURRENT_DATETIME
	INTERVAL	0 00:00:00.000
	TIME	CURRENT_TIME
Numeric	FLOAT	0.0
	DECIMAL	0
	INTEGER	0
	SMALLINT	0

If all columns in a table are not specified, the missing columns will be defined using the default values.

- Produces default mapping information that maps TurboIMAGE/XL data sets to ALLBASE/SQL tables and stores this information in the ATCINFO file (*DBEnvironmentName*CR). Specifically, mapping is done in the following areas:

  - **Data item and data set names**

    Some characters allowed in TurboIMAGE/XL names (specifically, + − * / ? ' % &) are not valid in ALLBASE/SQL names. Therefore, whenever the IMAGE/SQL utility encounters such a character in a TurboIMAGE/XL name, it converts it to an underscore (_).

  - **Data types**

    TurboIMAGE/XL data types are mapped to ALLBASE/SQL data types. When inexact or imprecise mapping is necessary, an I appears in the NOTES section of the DISPLAY MAP display. When a compound field is split into separate mapped columns, an S appears in the NOTES section of the display.

  - **User security**

    Initially, only the TurboIMAGE/XL database creator (DBC) is defined as a user in the DBEnvironment. For other users to access the attached database, the DBC must add users with the IMAGE/SQL utility's ADD USER command. Refer to Task 3, "Adding IMAGE/SQL Users," for more information.

- Once the database is attached, the DBC must add any additional IMAGE/SQL users.
- It is desirable to update data types (Task 4) and split mapped columns (Task 5) before IMAGE/SQL users access the attached database. This is because whenever a mapped column is split or the data type of a mapped column is updated, any user-created views containing these mapped columns are dropped.
- If the DBEnvironment does not exist, IMAGE/SQL automatically creates a DBEnvironment and DBEFiles. The default ATCINFO file name is the DBEnvironment name (up to 6 characters) appended by 'CR'.
- If you have a data set containing a compound item such that when it is mapped into an SQL name with suffix (-*n*, where *n* is an element number), and this mapped name happens to be the same as one of the existing mapped names within the same table, ATTACH will fail due to duplicate names. If this is the case, use a utility such as Adager or DBChange Plus to change the item name, then use the ATTACH command.

Task 3: Adding IMAGE/SQL Users

This task describes how to add IMAGE/SQL users.

Getting Ready

The DBC is the only IMAGE/SQL user defined when a database is attached. All other IMAGE/SQL users must be explicitly added. To perform this task, you must be the TurboIMAGE/XL DBC and the ALLBASE/SQL DBA. If you want to check the current users before adding new IMAGE/SQL users, issue the DISPLAY USER command. For example:

```
>>DISPLAY USER

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC

USER LOGON        DBOPEN MODE  USER PASSWORD  USER CLASS
----------        -----------  -------------  ----------

NANCY.ATC         5            ;              64
>>
```

Here, only the DBC is defined as an IMAGE/SQL user. Until additional IMAGE/SQL users are added, only the DBC can use IMAGE/SQL.

Performing the Task

Add IMAGE/SQL users with the ADD USER command. The class or password specified for the new user in this command must exist in the TurboIMAGE/XL database schema.

The following example adds the user SMITH.ATC to the DBEnvironment. It maps this ALLBASE/SQL DBEUserID to the user class 14 which is associated with password, CLERK, in the TurboIMAGE/XL schema.

```
>>ADD USER SMITH.ATC WITH CLASS=14, MODE=6

ALLBASE/SQL group SALES_14 created.
View SALES.DATE_MASTER_V14 created.
View SALES.CUSTOMER_V14 created.
View SALES.PRODUCT_V14 created.
View SALES.INVENTORY_V14 created.
View SALES.SALES_V14 created.
>>
```

The creation of these views allows SMITH.ATC to read only the data that is specified in the TurboIMAGE/XL schema for class 14 or password CLERK. The ADD USER command will accept the CLASS option or the PASS option. This user has DBOPEN mode 6 access to the database. The DISPLAY USER command now reflects the added user:

```
>>DISPLAY USER

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC

USER LOGON         DBOPEN MODE  USER PASSWORD  USER CLASS
----------         -----------  -------------  ----------

NANCY.ATC          5            ;              64
SMITH.ATC          6            CLERK          14

>>
```

Task Reference

- At attach time, the TurboIMAGE/XL database creator (DBC) is the only IMAGE/SQL user defined in the ATCINFO file (*DBEnvironmentName*CR).
- The ALLBASE/SQL syntax *User@Account* is used to add an IMAGE/SQL user. This name, referred to as the DBEUserID, is made up of an MPE/iX user and account name, connected with the @ symbol (or period). It must contain valid logon syntax.
- The ADD USER command maps the corresponding TurboIMAGE/XL user class to an ALLBASE/SQL group. The name of the new ALLBASE/SQL authorization group is *Ownername_nn*, where *nn* is the TurboIMAGE/XL user class number.
- To map the data set and data item security defined for the user class in the source TurboIMAGE/XL database schema, ALLBASE/SQL views are created for each new group. The name of these views is *MappedTableName_*V*nn*, where *nn* is the TurboIMAGE/XL user class number. One view is created for each mapped table the authorization group is allowed to read. Each view contains only those mapped columns to which the authorization group (user class) is allowed access.
- IMAGE/SQL supports all DBOPEN modes. Mode 5 is the default. Refer to the *TurboIMAGE/XL Database Management System Reference Manual* for more information about DBOPEN modes.

Task 4: Updating IMAGE/SQL Utility Data Type Mapping Information

This task describes how to select alternative IMAGE/SQL utility data type mapping.

Getting Ready

To select alternative data type mapping, you may first want to examine the default mapping done by the IMAGE/SQL utility. For example, the following default data type mapping information is stored in the ATCINFO file (*DBEnvironmentName*CR) for the mapped table CUSTOMER:

```
>>DISPLAY MAP CUSTOMER

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC
Owner Name       : SALES

MAPPED(SOURCE)   SOURCE           MAPPED            SOURCE     MAPPED
   TABLE         FIELD            COLUMN             TYPE       TYPE    NOTES
------------ ---------------- ------------------ ------- ----------- -----


CUSTOMER (CUSTOMER)
             ACCOUNT          ACCOUNT            J2      INTEGER
             LAST-NAME        LAST_NAME          X16     CHAR(16)
             FIRST-NAME       FIRST_NAME         X10     CHAR(10)
             INITIAL          INITIAL            U2      CHAR(2)
             STREET           STREET             X26     CHAR(26)
             CITY             CITY               X12     CHAR(12)
             STATE            STATE              X2      CHAR(2)
             ZIP              ZIP                X6      CHAR(6)
             CREDIT-RATING    CREDIT_RATING      R2      FLOAT       I

NOTES:
  I: Imprecise(float)/Incompatible(others) mapping between source and
     mapped data types
>>
```

By default, R2, the source data type of CREDIT_RATING, is mapped to a FLOAT. The I in the NOTES section indicates that this mapping may be imprecise because of differences in numeric storage between a 32-bit 3000 real (R2) and a 64-bit IEEE real (FLOAT).

Performing the Task

To change default data type mapping, use the UPDATE TYPE command. For example:

```
   >>UPDATE TYPE IN CUSTOMER.CREDIT_RATING TO CHAR(4)
   Updated information in table CUSTOMER.
   >>
```

In this example, CHAR(4) is specified as the data type mapping for CUSTOMER.CREDIT_RATING. The DISPLAY MAP command now reflects this change:

```
>>DISPLAY MAP CUSTOMER

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC
Owner Name       : SALES

MAPPED(SOURCE)   SOURCE            MAPPED       SOURCE     MAPPED
   TABLE         FIELD             COLUMN        TYPE       TYPE    NOTES
------------ ---------------- ------------------ ------- ----------- -----



CUSTOMER (CUSTOMER)
             ACCOUNT          ACCOUNT            J2      INTEGER
             LAST-NAME        LAST_NAME          X16     CHAR(16)
             FIRST-NAME       FIRST_NAME         X10     CHAR(10)
             INITIAL          INITIAL            U2      CHAR(2)
             STREET           STREET             X26     CHAR(26)
             CITY             CITY               X12     CHAR(12)
             STATE            STATE              X2      CHAR(2)
             ZIP              ZIP                X6      CHAR(6)
             CREDIT-RATING    CREDIT_RATING      R2      CHAR(4)    IU

NOTES:
  I: Imprecise(float)/Incompatible(others) mapping between source and
     mapped data types
  U: Source field has been updated
>>
```

The U in the NOTES section indicates that the data type mapping for this source field has been updated. The I indicates that the new mapping is incompatible with the source data type because numerical operations cannot be performed on character data.

Task Reference

- When the data type of a mapped column is updated, all user-created views based on IMAGE/SQL utility views containing the updated mapped column are dropped. Therefore, it is desirable to update data types before IMAGE/SQL users access the attached database.
- Table 2-6 summarizes IMAGE/SQL data type mapping defaults and alternatives. The following abbreviations and variables are used in Table 2-6:

MSB	most significant bit.
b	number of bytes needed for storage.
n	number of occurrences of the associated SQL type (the TurboIMAGE/XL sub-item length).

Table 2-6. IMAGE/SQL Data Type Mapping Defaults and Alternatives

Source Type	Bits	SQL Type Default	Comments on Default	SQL Type Alternative	Comments on Alternative
U*n*	8**n*	char(*n*)			
X*n*	8**n*	char(*n*)			
Z*n*†	8**n*	decimal(*n*,0)	default when *n* <= 15	char(*n*)††	default when *n* > 15
P*n*†	4**n*	decimal(*n-1*,0)	default when *n* <= 16	char(*n/2*)††	default when *n* > 16
I1, J1	16	smallint		decimal (5,*n*)	convert to decimal (5,*n*) where $0 \leq n < 5$
I2, J2	32	integer		decimal (10,*n*)	convert to decimal (10,*n*) where $0 \leq n < 10$
I3, J3	48	decimal(15,0)	value converted to packed decimal	char(6)††	8 bits binary data stored in each char
I4, J4†	64	char(20)††	value converted from binary to printable ASCII; zero-filled; sign included at run time	decimal(15,0) †††	value converted to packed decimal
				char(8)††	8 bits binary data stored in each char
K1†	16	integer	no loss of magnitude	smallint†††	MSB taken as sign

Table 2-6. IMAGE/SQL Data Type Mapping Defaults and Alternatives (continued)

Source Type	Bits	SQL Type Default	Comments on Default	SQL Type Alternative	Comments on Alternative
K2†	32	integer†††	MSB taken as sign	float	short IEEE converted to long IEEE
				decimal(15,0)	value converted to packed decimal
K3	48	decimal(15,0)	value converted to packed decimal	char(6)††	8 bits binary data stored in each char
K4†	64	char(20)††	value converted from binary to printable ASCII; zero-filled; no sign at run time	float	long IEEE assumed
				decimal(15,0) †††	value converted to packed decimal
				char(8)††	8 bits binary data stored in each char
K8	128	char(16)	value converted from binary to printable ASCII; zero-filled; no sign at run time	DATE, TIME, DATETIME, INTERVAL	value stored used to represent DATE, TIME, DATETIME, or INTERVAL
R2†	32	float	value converted from short HP 3000 real to long IEEE real at run time	char(4)††	8 bits binary data stored in each char; no IEEE conversion
R4†	64	float	value converted from HP 3000 real to long IEEE real at run time	char(8)††	8 bits binary data stored in each char; no IEEE conversion
X16	108	char(16)		DATE, DATETIME, INTERVAL, TIME	
all other data types†		char(*b*)††	8 bits binary data stored in each char		

† 100% mapping to an SQL type is not available.

†† Caution: When this type is mapped to char, the numeric meaning is lost in sorting, expressions, and aggregate functions.

††† Potential to under/overflow the available range. A run-time error results if the data value is outside the range of the SQL type. In this case, you may want to store the data in an alternative type.

Task 5: Splitting Mapped Columns

One data set field is sometimes used in TurboIMAGE/XL databases to hold several related units of data. This task describes how to easily access these individual data units by dividing them into separate mapped columns.

Getting Ready

Before splitting a mapped column, some preparation is necessary:

- Confirm that the TurboIMAGE/XL database containing the source field is attached to a DBEnvironment.
- Determine what TurboIMAGE/XL data type and length each individual unit would be assigned if it were to be defined as an individual data item in the TurboIMAGE/XL database schema.

  For example, PART-INFO, a large field in the INVENTORY data set, contains several units of information about a particular part. The DISPLAY MAP command shows how it is mapped at attach time:

  ```
  >>DISPLAY MAP FOR SALES INVENTORY.PART_INFO

  TurboIMAGE/XL DB : SALES.SERED.ATC
  DBEnvironment    : PARTSDBE.SERED.ATC
  Owner Name       : SALES

  MAPPED       SOURCE           MAPPED          SOURCE    MAPPED
  TABLE        FIELD            COLUMN          TYPE      TYPE      NOTES
  ------------ ---------------- --------------- --------- --------- -----

  INVENTORY
               PART-INFO        PART_INFO       X60       CHAR(60)
  >>
  ```

  Specifically, PART-INFO contains:

  - □ A part identification code (the first 4 bytes of PART-INFO).
  - □ The version number of the part (the next 2 bytes of PART-INFO).
  - □ Brief notes about the part (the last 54 bytes of PART-INFO).

  Each unit of information corresponds to the following TurboIMAGE/XL data types:

  - □ The part identification code is X4 (4 bytes).
  - □ The version number of the part is I1 (2 bytes).
  - □ The notes about the part is X54 (54 bytes).

- Make sure that the sum of the lengths of the data units calculated in step 2 matches the length of the original field, in this case PART-INFO (60 bytes). (Here 4 + 2 + 54 = 60, so the lengths correspond.)

- Decide what to name the new mapped columns. For example, the new mapped columns in the mapped table INVENTORY are named:

  PART_ID_CODE
  PART_VERSION_NO
  PART_NOTES

- Determine what SQL data type(s) can be mapped to each TurboIMAGE/XL data type identified in step 2; refer to Table 2-6. When alternative data type mapping possibilities exist, decide which mapping best reflects the format of the data.

Performing the Task

The SPLIT command requires you to enter the name of the new mapped column, its equivalent TurboIMAGE/XL data type, and, optionally, its SQL data type. Note that the ampersand (&) is used to continue to the next line. Be sure to list the new mapped columns in the order in which they are stored in the original source field:

For example:

```
>>SPLIT INVENTORY.PART_INFO INTO PART_ID_CODE:X4:CHAR(4),&
                                 PART_VERSION_NO:I1:SMALLINT,&
                                 PART_NOTES:X54:CHAR(54)

Updated information in table INVENTORY.
```

Here the mapped column PART_INFO in the mapped table INVENTORY is being split into three new mapped columns:

PART_ID_CODE of type CHAR(4)
PART_VERSION_NO of type SMALLINT
PART_NOTES of type CHAR(54)

Note that for clarity, the SQL data types are explicitly specified in the above example. However, because they represent default data type mapping, it is not mandatory to explicitly specify these SQL data types.

The DISPLAY MAP command shows the newly split columns:

```
>>DISPLAY MAP FOR SALES INVENTORY

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC
Owner Name       : SALES

MAPPED       SOURCE          MAPPED             SOURCE  MAPPED
TABLE        FIELD           COLUMN             TYPE    TYPE        NOTES
------------ --------------- ------------------ ------- ----------- -----

INVENTORY
             PRODUCT#        PRODUCT#           U8      CHAR(8)
               .               .                 .        .
               .               .                 .        .
               .               .                 .        .

             LOCATION-BIN    LOCATION_BIN       Z2      DECIMAL(2,0) I
     new ==> PART-INFO       PART_ID_CODE       X4      CHAR(4)      S
     new ==> PART-INFO       PART_VERSION_NO    I1      SMALLINT     S
     new ==> PART-INFO       PART_NOTES         X54     CHAR(54)     S

NOTES:
  I:  Imprecise(float)/Incompatible(others) mapping between source and
      mapped data types
  S:  Source field has been split.
>>
```

The S in the NOTES section indicates that the source field, PART-INFO, has been split into separate mapped columns.

Task Reference

- Refer to Table 2-6 for alternative data type mapping information.
- SQL names can be up to 20 bytes in length and can be made up of any combination of letters (A to Z), decimal digits (0 to 9), $, #, @, or underscore (_). The first character cannot be an underscore or a decimal digit.
- It is desirable to split mapped columns before IMAGE/SQL users access the attached database. This is because when a mapped column is split, any user-created views containing the mapped column to be split are dropped.

Task 6: Updating Information about IMAGE/SQL Users

This task describes how to update IMAGE/SQL user information.

Getting Ready

To perform this task, you must be both the TurboIMAGE/XL DBC and a DBA of the attached DBEnvironment.

If you want to check the current users before adding new IMAGE/SQL users, issue the DISPLAY USER command. For example:

```
>>DISPLAY USER

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC

USER LOGON         DBOPEN MODE  USER PASSWORD  USER CLASS
----------         -----------  -------------  ----------

NANCY.ATC          5            ;              64
SMITH.ATC          6            CLERK          14

>>
```

Performing the Task

Use the UPDATE USER command to change the user password, user class, or DBOPEN mode associated with an IMAGE/SQL user.

For example:

```
>>UPDATE USER SMITH.ATC TO MODE=5
>>
```

Here, SMITH.ATC has DBOPEN mode 5 access to the database. Note that because the PASS or CLASS parameter was not specified, the user class does not change.

The DISPLAY USER command now reflects the updated information:

```
>>DISPLAY USER

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC

USER LOGON         DBOPEN MODE  USER PASSWORD  USER CLASS
----------         -----------  -------------  ----------

NANCY.ATC          5            ;              64
SMITH.ATC          5            CLERK          14

>>
```

Task Reference

- The SQL syntax *User@Account* (or *User.Account*) is used to add an IMAGE/SQL user. This name, referred to as the DBEUserID, is made up of an MPE/iX user and account name, connected with the @ symbol (or period). It must contain valid logon syntax.
- The PASS= or CLASS= parameter of the UPDATE USER command is used to change the password or class associated with the IMAGE/SQL user. The password or class specified must exist in the TurboIMAGE/XL database schema.

Task 7: Deleting IMAGE/SQL Users

This task describes how to delete IMAGE/SQL users.

Getting Ready

To perform this task, you must be both the TurboIMAGE/XL DBC and an SQL DBA.

If you want to check the current users before adding new IMAGE/SQL users, issue the DISPLAY USER command. For example:

```
>>DISPLAY USER

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC

USER LOGON         DBOPEN MODE  USER PASSWORD  USER CLASS
----------         -----------  -------------  ----------

NANCY.ATC          5            ;              64
SMITH.ATC          5            CLERK          14

>>
```

Performing the Task

Use the DELETE USER command to delete an IMAGE/SQL user. For example:

```
>>DELETE USER SMITH.ATC
>>
```

Here, SMITH.ATC is deleted as an IMAGE/SQL user and removed from the corresponding authorization group.

The DISPLAY USER command now reflects the deletion.

```
>>DISPLAY USER

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC

USER LOGON         DBOPEN MODE  USER PASSWORD  USER CLASS
----------         -----------  -------------  ----------

NANCY.ATC          5            ;              64

>>
```

Note that SMITH.ATC is still configured as an SQL user because ALLBASE/SQL tables and views may be owned by this *User.Logon*.

Task Reference

- When the last SQL user logon associated with a TurboIMAGE/XL user class is deleted, the SQL authorization group (*OwnerName_UserClass#*) and views associated with this user class are not dropped. Doing so would invalidate any views that may have been created based on these IMAGE/SQL views.
- The SQL syntax *User@Account* (or *User.Account*) is used to add an IMAGE/SQL user. This name, referred to as the DBEUserID, is made up of an MPE/iX user and account name, connected with the @ symbol (or period). It must contain valid logon syntax.

Task 8: Adding a Log File

This task describes how to add a new log file to the DBEnvironment.

Performing the Task

Run SQLUtil, issue the ADDLOG command, and add a new log file as in the following example:

```
:RUN SQLUTIL.PUB.SYS
>>ADDLOG
DBEnvironment Name: PARTSDBE
Maintenance Word: (Return)
Enter Log File Name(s) Separated by a Blank? LGN1
New Log File Size? 300
Add Log File (y/n)? y

Log file 'LGN1' was Added.
Log Identifier Is: 2
>>exit
```

Task Reference

- There are several reasons to add new log file(s) of the size and number required:
  - To accommodate the largest transaction carried out by the maximum number of concurrent users.
  - To take action to prevent *Log Full* messages.
  - To control the frequency of checkpoints.
- The DBEnvironment can be in use when this command is executed.
- Add log files one at a time using separate ADDLOG commands.
- The maximum number of log files in a DBEnvironment is 34.

Task 9: Detaching a TurboIMAGE/XL Database

This task describes how to detach a TurboIMAGE/XL database from a DBEnvironment.

Getting Ready

Before detaching a TurboIMAGE/XL database, do the following:

- Obtain exclusive access to the DBEnvironment before detaching a TurboIMAGE/XL database.
- Make sure the appropriate SET SQLDBE and SET TURBODB commands have been issued. To check the status of these commands, use the DISPLAY OPTIONS command:

```
>>DISPLAY OPTIONS
Current base      :
Current SQLDBE    :
Echo              : ON
Command Logging   : ON
Current Log File  : ATCLOG.SERED.ATC
>>
```

- If necessary, issue the SET commands. For example:

```
>>SET SQLDBE PARTSDBE.SERED.ATC
>>SET TURBODB SALES.SERED.ATC
>>
```

Performing the Task

Once you have issued the appropriate SET commands, you are ready to detach the database from the DBEnvironment. For example:

```
>>DETACH
>>
```

This command detaches the SALES database from the PartsDBE DBEnvironment. If SALES is the only database attached to PartsDBE, the ATCINFO file (*DBEnvironmentName*CR) will be deleted. If PartsDBE is the only DBEnvironment to which SALES is attached, the SALESTC file will also be deleted.

Task Reference

- There are several reasons to detach a TurboIMAGE/XL database from a DBEnvironment:
  - If the database is to be restructured. (This includes restructuring with DBChange and other third-party restructuring tools.)
  - If you want to reset all mapping information to default values.
- All mapped tables, all IMAGE/SQL views based on these tables, and all user-created views based on IMAGE/SQL views and tables are dropped when the database is detached.
- The ATCINFO file (default name *DBEnvironmentName*CR) is purged when it no longer contains mapping information about any databases.
- The *DBaseName*TC file is deleted when the database is no longer attached to any DBEnvironments.

Task 10: Displaying IMAGE/SQL Utility Information

This task describes how to display IMAGE/SQL information using several IMAGE/SQL utility commands. These commands include:

DISPLAY OPTIONS	Displays the options in effect for your current IMAGE/SQL utility session.
DISPLAY TURBODB	Displays all the TurboIMAGE/XL databases associated with a specific DBEnvironment.
DISPLAY SQLDBE	Displays all the DBEnvironments associated with a specific TurboIMAGE/XL database.
DISPLAY MAP	Displays the current data type mapping information for a specific TurboIMAGE/XL database.
DISPLAY USER	Displays the current information about users in a specific TurboIMAGE/XL database.
HELP	Provides the syntax of IMAGE/SQL utility commands.

Getting Ready

- The display commands provide two options: either issue SET commands before displaying information or specify a particular TurboIMAGE/XL database or DBEnvironment as part of the display command.
- For DISPLAY SQLDBE and DISPLAY TURBODB to display useful information, at least one database should be attached.
- For DISPLAY MAP AND DISPLAY USER to display useful information, a database should be attached.

Performing the Task

Two examples using the DISPLAY commands are shown below and on the next page.

Example 1: Displaying Database Information

To see all the TurboIMAGE/XL databases and mapped tables currently associated with PartsDBE, use the DISPLAY TURBODB command.

```
:RUN IMAGESQL.PUB.SYS
>>DISPLAY TURBODB TABLES FOR PARTSDBE.SERED.ATC


DBEnvironment     : PARTSDBE.SERED.ATC

TURBOIMAGE/XL DATABASE   OWNER           MAPPED TABLE
----------------------   -----           ------------

SALES.SERED.ATC          SALES           DATE_MASTER
                                         CUSTOMER
                                         PRODUCT
                                         VENDOR
                                         INVENTORY
                                         SALES

Total Databases : 1
>>
```

Any databases attached to the DBEnvironment and their associated mapped tables are displayed.

Example 2: Displaying Database Mapping Information

The DISPLAY MAP command shows how TurboIMAGE/XL data sets are mapped to SQL tables. In the following example, information about the mapped table INVENTORY is displayed.

```
>>DISPLAY MAP INVENTORY

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC
Owner Name       : SALES

MAPPED(SOURCE)   SOURCE             MAPPED       SOURCE     MAPPED
   TABLE         FIELD              COLUMN        TYPE       TYPE    NOTES
------------ ---------------- ------------------ ------- ----------- -----



INVENTORY (INVENTORY)
             PRODUCT#         PRODUCT#           U8      CHAR(8)
             ON-HAND-QTY      ON_HAND_QTY        J2      INTEGER
             VENDOR           VENDOR             X16     CHAR(16)
             OTHER-VENDORS    OTHER-VENDORS_1    X16     CHAR(16)    S
             OTHER-VENDORS    OTHER-VENDORS_2    X16     CHAR(16)    S
             OTHER-VENDORS    OTHER-VENDORS_3    X16     CHAR(16)    S
             UNIT-COST        UNIT_COST          P8      DECIMAL(7,0)
             LAST-SHIP-DATE   LAST_SHIP_DATE     X6      CHAR(6)
             LOCATION-BIN     LOCATION_BIN       Z2      DECIMAL(2,0)
             PART-INFO        PART_INFO          X60     CHAR(60)

NOTES:
  S: Source field has been split
```

Here, the S in the NOTES section indicates that the compound data item, OTHER_VENDORS, has been mapped and split into three mapped columns.

Task 11: Issuing MPE/iX Commands from the IMAGE/SQL Utility

This task describes how to issue MPE/iX commands from the IMAGE/SQL utility.

Performing the Task

To issue an MPE/iX command from the IMAGE/SQL utility, enter a colon (:) and the name of the MPE/iX command you want to issue.

For example, to issue the LISTF command:

```
>>:LISTF

FILENAME

ATCINFO     ATCUPROG    ATCUT000    ATCUTCAT    ATCUTIL     CREASQL
DBDUCB00    DBDUCB01    DBDUCB02    DBDUCB03    DBMON       DHSQLCT
ISQL        LOG1        MMXL        ORDERDF1    SALES       SALES01
SALES02     SALES03     SALES04     SALES05     SALES06     SALESTC
ORDERXF1    PARTSDBE    PARTSF0     PARTSLG1    PARTSLG2    PURCHDF1
PURCHXF1    READSQL     RECDF1      WAREHDF1    WAREHXF1

>>
```

The IMAGE/SQL utility prompt returns after the MPE/iX command is executed.

Task 12: Setting IMAGE/SQL File Equations

This task describes how to set IMAGE/SQL utility file equations for two files: ATCINFO and ATCLOG.

Performing the Task

Setting a File Equation for ATCINFO

An ATCINFO file equation can only be set before the file is created (before any TurboIMAGE/XL databases are attached to the DBEnvironment). Only the file name can be specified. Other parameters of the FILE command will not be valid at attach time.

For example:

```
:FILE ATCINFO = PARTSACR
:
```

Setting a File Equation for ATCLOG

A file equation for ATCLOG can be issued before beginning an IMAGE/SQL utility session.

For example:

```
:FILE ATCLOG = SALELOG1; SAVE
:
```

In the above example, when the logging option is on, IMAGE/SQL utility commands are logged to the permanent file SALELOG1.

A file equation for ATCLOG can also be set or reset from within the IMAGE/SQL utility.

For example:

```
>>:FILE ATCLOG = SALELOG2; SAVE
>>
```

Now when the logging option is on, IMAGE/SQL utility commands are logged to the permanent file SALELOG2.

Task Reference

ATCINFO Reference

- The ATCINFO file is a permanent privileged file containing mapping information about data set and field names, data types, and user security. Its formal file designator is ATCINFO. The default name of the ATCINFO file is *DBEnvironmentName*CR.
- One ATCINFO file exists for each DBEnvironment. It is created in the same group and account as the DBEnvironment and is considered part of the DBEnvironment.
- The ATCINFO file equation can only be used to specify a different file name. It cannot be used to override other ATCINFO file characteristics.

ATCLOG Reference

- ATCLOG is a temporary unnumbered ASCII file to which all IMAGE/SQL utility commands are written when the IMAGE/SQL utility logging is on. If this file does not already exist, it is created. If it already exists as a temporary or permanent file, log file records are appended to it. Its formal file designator is ATCLOG.
- When an IMAGE/SQL utility session begins, logging is on and the ATCLOG file is created (if it does not already exist) and opened in the user's logon group and account. By default, it is a temporary file named ATCLOG, but you may want to set a file equation for this file.
- If you want to save the temporary file, either specify SAVE as a part of the FILE command or save the file before ending your current MPE/iX session.
- The FILE command can be used to override default ATCLOG file attributes, such as size and file domain. However, the log file must remain an unnumbered ASCII file.

Task 13: Logging IMAGE/SQL Utility Commands

This task describes how to log IMAGE/SQL utility commands and how to save frequently used command sequences in different log files so they can easily be reissued in batch or interactive mode.

Performing the Task

By default, IMAGE/SQL utility commands are logged to the ASCII file ATCLOG, which you can read and edit. You can change log files within the IMAGE/SQL utility by issuing a file equation for ATCLOG. For example, to rename and save the IMAGE/SQL utility log file as a permanent file, set a file equation for the ATCLOG file before the

IMAGE/SQL utility is run:

```
:FILE ATCLOG = DOATTACH; SAVE
:
```

When the IMAGE/SQL utility is run, the DISPLAY OPTIONS command confirms that commands are being logged to DOATTACH.

```
>>DISPLAY OPTIONS
Current base        :
Current SQLDBE      :
Echo                : ON
Command Logging     : ON
Current Log File    : DOATTACH.SERED.ATC

*** Database is not attached.
>>
```

The SALES database is then attached to the PartsDBE DBEnvironment:

```
>>SET TURBODB SALES.SERED.ATC
>>SET SQLDBE PARTSDBE.SERED.ATC
>>ATTACH
>>
```

After the attach, the contents of the log file DOATTACH look like this:

```
DISPLAY OPTIONS
SET TURBODB SALES.SERED.ATC
SET SQLDBE PARTSDBE.SERED.ATC
ATTACH
```

To close this log file and write to a new one, issue another MPE/iX FILE command from within the IMAGE/SQL utility. For example:

```
>>:FILE ATCLOG=UPDATYPE;SAVE
>>
```

The DISPLAY OPTIONS command confirms that commands are now being logged to UPDATYPE:

```
>>DISPLAY OPTIONS
Set Turbodb        : SALES.SERED.ATC
Set Sqldbe         : PARTSDBE.SERED.ATC
Echo               : ON
Command Logging    : ON
Current Log File   : UPDATYPE.SERED.ATC

*** Database is attached.
```

Any IMAGE/SQL utility commands now issued are written to UPDATYPE. For example, when the following command is issued,

```
>> UPDATE TYPE IN CUSTOMER.CREDIT_RATING TO CHAR(4)
Updated information in table CUSTOMER.
>>
```

the log file UPDATYPE contains the following commands:

```
DISPLAY OPTIONS
UPDATE TYPE IN CUSTOMER.CREDIT_RATING TO CHAR(4)
```

Using this technique, you can create several files containing often-used IMAGE/SQL utility commands. The commands in these files can then be executed using the XEQ command.

Log files can be edited with a text editor. (Be sure to keep the edited file unnumbered or it cannot be used as a command file.)

Note Many editors automatically insert an end-of-file marker at the end of the text in an edited file. This prevents the file from having log records appended to it. For this reason, if the original file is to be used as a log file, you should keep the edited file under a different name.

Task Reference

- Task 12, "Setting IMAGE/SQL Utility File Equations," contains more information about issuing IMAGE/SQL utility file equations.
- Task 14, "Using IMAGE/SQL Utility Command Files," contains more information about using the IMAGE/SQL utility XEQ command interactively or in batch to execute IMAGE/SQL utility commands.
- When an IMAGE/SQL utility session begins, logging is on. The temporary file ATCLOG is created (if it does not already exist) and opened in the user's logon group and account. If the log file already exists as a temporary or permanent file, log records are appended to it.
- If you want to save the temporary file, either save the file before ending your current MPE/iX session or specify ;SAVE as a part of the FILE equation.
- The FILE command can be used to override default ATCLOG file attributes, such as size and file domain. However, the log file must remain an unnumbered ASCII file.

Task 14: Using IMAGE/SQL Utility Command Files

This task describes how to execute an IMAGE/SQL utility command file in interactive or batch mode.

Getting Ready

An IMAGE/SQL utility command file can be created with a text editor or can be produced as part of the logging process. If you create or edit this file with an editor, it must be kept unnumbered.

In the following examples, the file UPDATYPE contains the following commands:

```
    SET TURBODB SALES.SERED.ATC
    SET SQLDBE PARTSDBE.SERED.ATC
    UPDATE TYPE IN CUSTOMER.CREDIT_RATING TO CHAR(4)
    EXIT
```

Note that the commands in this command file assume the TurboIMAGE/XL database is already attached to the DBEnvironment.

If you want to see commands and comments as the command file is executed, make sure the ECHO option is on.

Performing the Task

The XEQ command allows you to specify a file containing IMAGE/SQL utility commands as its parameter. To interactively execute commands listed in a command file, run the IMAGE/SQL utility by typing RUN IMAGESQL.PUB.SYS, then issue the XEQ command. Note that in this example ECHO is off.

```
:RUN IMAGESQL.PUB.SYS

HP36385 B.FO.10        IMAGE/SQL Utility       FRI, DEC 18, 1992, 11:30 AM
(C) COPYRIGHT HEWLETT-PACKARD COMPANY 1992

>>XEQ UPDATYPE
Updated information in table CUSTOMER.
:
```

The SALES database is now attached to the PartsDBE DBEnvironment and data type mapping information has been updated for the CUSTOMER.CREDIT_RATING column of the INVENTORY table.

You can also issue XEQ commands in batch mode. The following job stream file contains XEQ commands that execute the commands in the UPDATYPE file.

```
!job JIMSQL,NANCY/KEVIN.ATC/MGR,SERED/ALL
!comment**************************************************************
!comment*      This job executes an IMAGESQL command file.
!comment**************************************************************
!
!tell NANCY.ATC;   /-->Start JIMSQL for SALES
!
!run IMAGESQL.PUB.SYS
!
!comment**************************************************************
!comment*      The UPDATYPE command file contains commands that SET
!comment*      the SALES database and the PartsDBE DBEnvironment.
!comment*      It then specifies alternative data type mapping for source
!comment*      data set fields and exits the IMAGE/SQL utility.
!comment**************************************************************
!
XEQ UPDATYPE
!
!tell NANCY.ATC;   /-->End JIMSQL for SALES
!
!eoj
```

If an error occurs in batch mode, the job terminates. The remaining commands are flushed.

Task Reference

- An IMAGE/SQL command file is an unnumbered file containing a list of IMAGE/SQL commands. If commands span more than one line, use an ampersand (&) to continue the command to the next line.
- Task 13, "Logging IMAGE/SQL Utility Commands," shows how to use the IMAGE/SQL utility logging facility to create and save files containing often-issued IMAGE/SQL utility commands.
- For the syntax of the ECHO command, refer to Chapter 4, "IMAGE/SQL Utility Commands."

Task 15: Selecting TurboIMAGE/XL Data with SQL

This task explains how IMAGE/SQL users select TurboIMAGE/XL data with SQL.

Getting Ready

To successfully select TurboIMAGE/XL data with IMAGE/SQL, users need to know the following:

- How to use their available SQL interface. The examples in this manual use ISQL, which also requires familiarity with the SQL SELECT statement.
- The names of the mapped tables and/or views to which they have access.
- Which columns map to TurboIMAGE/XL search items, key items (with or without B-Tree indices), and items on which third-party indices exist. Under certain circumstances, using these mapped columns when selecting data can improve performance (see Task Reference).
- Which data is of type FLOAT. When selecting this data, users should specify a range of values rather than a particular number. This is necessary because some precision is lost when converting to FLOAT. Also, a decimal point must be included in the value for better performance.

Performing the Task

In the example below, a CONNECT statement for PartsDBE is issued from the ISQL prompt. A select statement then retrieves all the data from the view SALES.VENDOR_V13.

```
isql=> CONNECT TO 'PartsDBE';
isql=> SELECT * FROM SALES.VENDOR_V13;

select * from sales.vendor_v13;
----------------+-------------------------+------------+-----+------
VENDOR          |STREET                   |CITY        |STATE|ZIP
----------------+-------------------------+------------+-----+------
Celtic Graphics |105 19th Ave.            |Seattle     |WA   |98115
Trident 3D      |55 Homestead Road        |Cupertino   |CA   |95014
Ablrn Tech.     |90 Marina Way            |Berkeley    |CA   |94708
Space Ent.      |110 Homestead Ave.       |Cupertino   |CA   |95014
Cutler Micro    |9442 E. 57th Ave.        |Seattle     |WA   |98115
Seminational Co.|5000 Marina Way          |San Diego   |CA   |92093
.                 .                         .           .      .
.                 .                         .           .      .
```

Users can also join data from more than one view. The query in the following example retrieves data from two views, both with a column containing product numbers. The product number, the quantity, and the name of an alternative vendor is selected when the product number appears in both views.

Specifically, the query selects the following columns:

- OTHER_VENDORS_1 from view SALES.INVENTORY_14
- QUANTITY from view SALES.SALES_V14
- SALES.SALES.PRODUCT# from view SALES.SALES_V14

Note that to eliminate ambiguity, because PRODUCT# exists in both views, the fully qualified column name must always be specified.

```
isql=> SELECT OTHER_VENDORS_1, QUANTITY, SALES.SALESV_14.PRODUCT#
>      FROM SALES.INVENTORY_V14, SALES.SALES_V14
>      WHERE SALES.INVENTORY_V14.PRODUCT#=SALES.SALES_V14.PRODUCT#;

select other_vendors_1, quantity, sales.salesv_14.product# from sales.inv..

----------------+---------+--------
OTHER_VENDORS_1 |QUANTITY |PRODUCT#
----------------+---------+--------
Ablrn Tech.     |        4|P4943
Celtic Graphics |        2|P6644
Celtic Graphics |    10050|P3523
.                    .        .
.                    .        .
```

In this example, whenever the product numbers in the two views match, ISQL displays columns PRODUCT#, OTHER_VENDORS_1, and QUANTITY.

Task Reference

- The structure of the TurboIMAGE/XL database cannot be changed with IMAGE/SQL commands. Therefore, SQL statements that alter the structure of the database are not available to IMAGE/SQL users.
- When users have access to the entire data entry, they can select data from the table itself. If they do not have access to the entire data entry, they must select data from a view of the table created for them by IMAGE/SQL. Table names are of the form *OwnerName.MappedTableName*. View names are of the form *OwnerName.MappedTableName*_V *UserClass#*.

- In WHERE clause, specifying columns that map to TurboIMAGE/XL search items, key items (with or without B-Tree indices), or items that have third-party indices will improve performance under the following conditions:
  - The WHERE clause compares a mapped column and a value for equality:

    ```
    isql=> SELECT * FROM SALES.SALES_V11
    >      WHERE PRODUCT# = '235'
    > ...
    ```

    When a column maps to an item which has a B-Tree or third-party index, an operator, other than equality, can also be used (such as > or >=).
  - The WHERE clause has more than one expression, each containing a different mapped column. These subexpressions are connected with the AND operator.

    ```
    isql=> SELECT * FROM SALES.SALES_V11
    >      WHERE (product# = '234')
    >        AND (purchased_date = '032189')
    > ...
    ```
  - The WHERE clause has more than one expression, each containing the same mapped column. These subexpressions either use the IN operator or are connected with the OR operator.

    ```
    isql=> SELECT * FROM SALES.SALES_V11
    >      WHERE (product# IN ('224', '321'))
    > ...
    ```

Task 16: Maintaining the ATCINFO File

This task describes how to maintain the ATCINFO file.

Getting Ready

Maintenance for the ATCINFO file (*DBEnvironmentName*CR) may be necessary in either of the following situations:

- If a crash occurs while the ATCINFO file is being modified. This may be the case if a crash takes place when the IMAGE/SQL administrator is in the midst of an IMAGE/SQL utility command that updates the ATCINFO file. When this occurs, the RECOVER command can be used to reconstruct the ATCINFO file.
- If the ATCINFO file contains too much free space. This may be the case if multiple databases are attached and then some are detached from the DBEnvironment. When this occurs, the RECOVER command can be used to compact the data in the ATCINFO file and return the free space to the file system.

Performing the Task

To recover the ATCINFO file associated with the PartsDBE DBEnvironment, issue the following commands:

```
:RUN IMAGESQL.PUB.SYS
>>SET SQLDBE PartsDBE
>>RECOVER
Checking physical file consistency and recovering free space.
Deleting unused mapped table entries.
Checking external cross references.
>>
```

Task Reference

- For the syntax of the RECOVER command, refer to Chapter 4, "IMAGE/SQL Utility Commands."

3

Understanding IMAGE/SQL

This chapter explains how IMAGE/SQL works and outlines the SQL functionality available to IMAGE/SQL users.

This chapter describes:

- Files created by IMAGE/SQL
- Events that take place during an attach
- Events that take place during a detach
- IMAGE/SQL security
- IMAGE/SQL data type mapping
- Run-time events
- Mapped table access with SQL

How IMAGE/SQL Works

IMAGE/SQL users can select data in mapped tables in the same way that data in ALLBASE/SQL tables is selected.

To understand how this is accomplished, you need to know:

- What files are used by IMAGE/SQL
- How IMAGE/SQL files are used
- What takes place during the attach/detach process
- How IMAGE/SQL maps TurboIMAGE/XL security
- How IMAGE/SQL maps TurboIMAGE/XL data types
- What takes place at run time

IMAGE/SQL Files

IMAGE/SQL creates two files: the ATCINFO file with a default name of *DBEnvironmentName*CR in the same group and account as the DBEnvironment and *DBName*TC in the same group and account as the TurboIMAGE/XL database. This is shown in Figure 3-1.

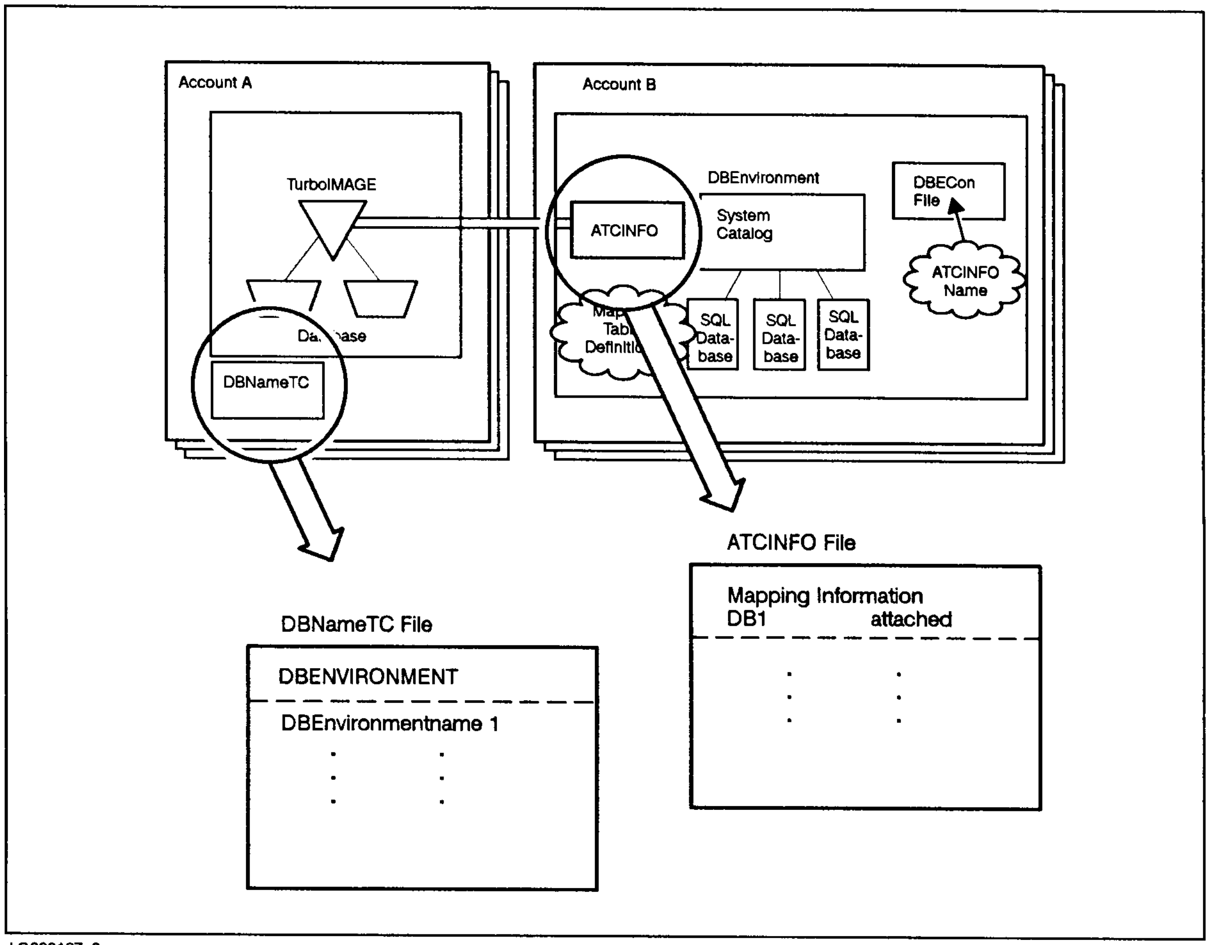


Figure 3-1. Files Created by IMAGE/SQL

These files contain information about the relationships between all attached TurboIMAGE/XL databases and their DBEnvironments. Specifically:

ATCINFO is a permanent privileged file in the same group and account as the DBEnvironment. It contains mapping information about each TurboIMAGE/XL database attached to the DBEnvironment. One ATCINFO file exists for each DBEnvironment.

It is named *DBEnvironmentName*CR where *DBEnvironmentName* is up to six characters of the actual DBEnvironmentName unless a file equation has been set before the attach. The ATCINFO filename is placed in the DBECon file of the DBEnvironment so that it can be located whenever IMAGE/SQL needs to use or update the information in this file.

DBNameTC is a permanent privileged file in the same group and account as the TurboIMAGE/XL database. It contains the fully qualified names of the DBEnvironments to which the TurboIMAGE/XL database is attached. This information is used to let external utilities such as DBUTIL know that the database is attached to one or more DBEnvironments.

What Takes Place During an Attach?

An attach is the process that establishes the connection between the TurboIMAGE/XL database and the DBEnvironment. Figure 3-2 shows a TurboIMAGE/XL database and DBEnvironment before the TurboIMAGE/XL database is attached.

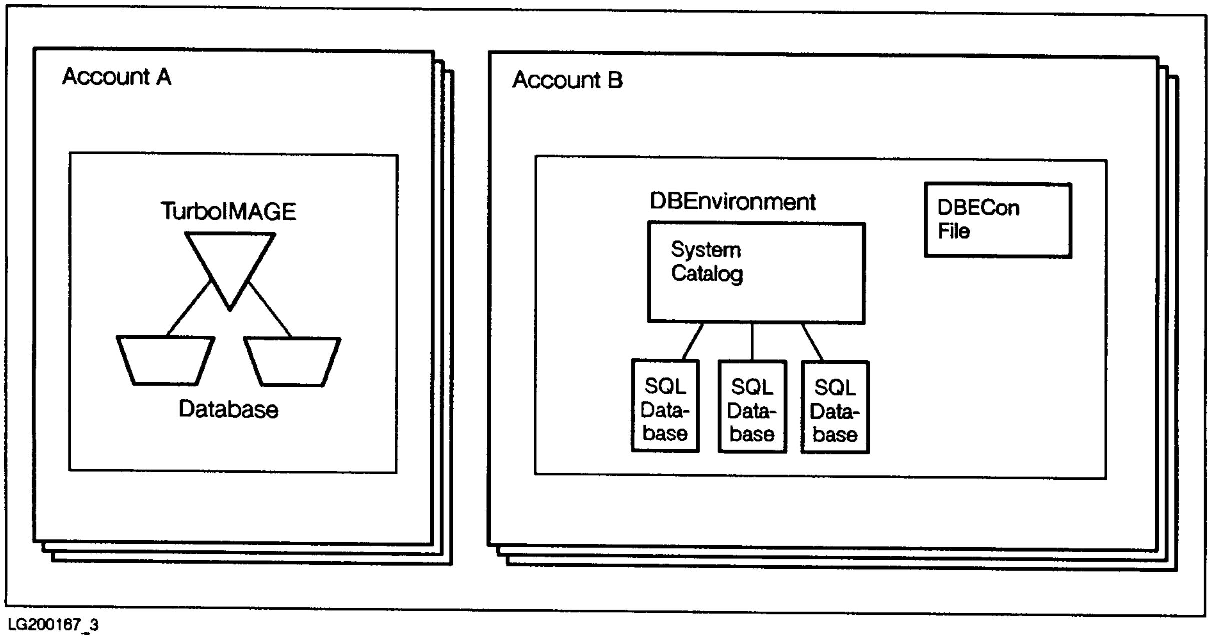


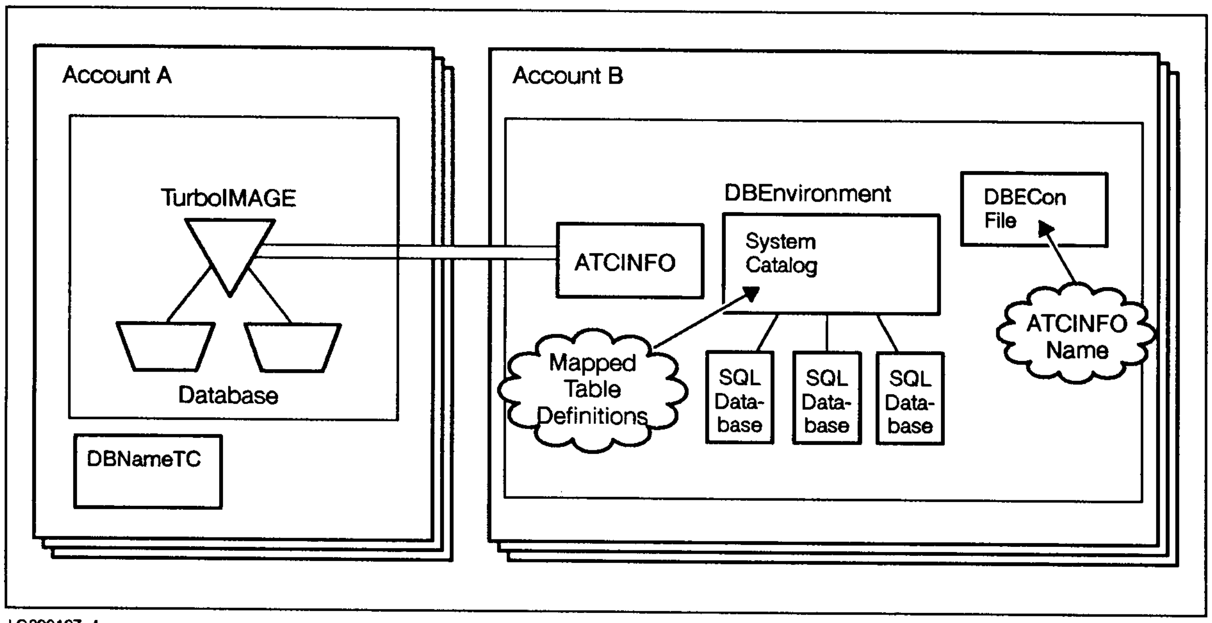
Figure 3-2. A TurboIMAGE/XL Database and a DBEnvironment Before the Attach

When the TurboIMAGE/XL database is attached to the DBEnvironment, several events take place:

1. The ATCINFO (*DBEnvironmentName*CR) and *DBName*TC files are created.
2. Definitions of the TurboIMAGE/XL mapped tables are placed in the system catalog of the DBEnvironment. These entries in the system catalog identify the tables as mapped tables. The naming convention for SQL tables is *OwnerName.TableName*. By default, in mapped tables, IMAGE/SQL specifies the database name as the owner name and the data set name as the table name. Thus, the naming convention for mapped tables is *MappedDatabaseName.MappedDataSetName*. (At attach time, you must substitute a different owner name if an already attached database has the same name.)
3. Definitions of the TurboIMAGE/XL mapped columns are placed in the system catalog of the DBEnvironment.
4. Definitions for hash indices on TurboIMAGE/XL master key item and detail search items are entered into the system catalog for the DBEnvironment. All master keys, except P and Z data types, are entered as unique hash indices. Definitions for detail search items, as well as master P and Z key types, are entered as non-unique hash indices. These definitions are in the views SYSTEM.IMAGEKEY and CATALOG.IMAGEKEY. With the hash indices, the performance at run-time is much better when the equality operator is used with the SQL statements.

   If the B-Tree index is created for the key item of a master data set using TurboIMAGE/XL, a definition for a unique B-Tree index is placed in the system catalog. In addition, definitions for non-unique B-Tree indices for all of the search items of the detail data sets to which this key item has a path are added in the system catalog. The definitions for these B-Tree indices are in the views SYSTEM.INDEX and CATALOG.INDEX of the system catalog.
5. If the database is enabled for third-party indexing, definitions for these indices are also added to the system catalog of the DBEnvironment, except for keyword indices and those indices for which the third-party provides no information. These definitions may be for unique or non-unique indices. The definitions are in the views SYSTEM.TPINDEX and CATALOG.TPINDEX of the system catalog. The definitions for third-party indices are also placed in the system catalogs of other DBEnvironments to which the database is attached.
6. The TurboIMAGE/XL database creator (DBC) is defined in the ATCINFO file as an IMAGE/SQL user. For security reasons, all other IMAGE/SQL users must be explicitly added by the DBC.

Figure 3-3 shows an attached database.



LG200167_4

Figure 3-3. An Attached TurboIMAGE/XL Database

What Takes Place During a Detach?

When a database is detached, you can no longer use SQL to access TurboIMAGE/XL data. All views based on mapped tables are dropped. This includes views created by users, as well as IMAGE/SQL-created views. Mapped table definitions are removed from the system catalog and all mapping information about the detached TurboIMAGE/XL database is removed from the ATCINFO file. This results in the removal of definitions of hash, B-Tree, and third-party indices from the system catalog.

If you detach the only TurboIMAGE/XL database attached to the DBEnvironment, the ATCINFO file is deleted, and its name is removed from the DBECon file. If you detach the TurboIMAGE/XL database from the only DBEnvironment it is attached to, the *DBName*TC file is deleted.

If you plan to restructure the database, or if you want to remove all mapping information about the database from the ATCINFO file, detach the database. If you only want to change the capacity of a data set, you do not need to detach the database.

When the database is detached, all customized mapping information (alternative data types and added IMAGE/SQL users) is lost and must be remapped when the database is reattached. IMAGE/SQL utility command files are useful for this purpose (refer to Tasks 13 and 14 in Chapter 2).

To protect database security, be sure command files that contain passwords and other sensitive information are carefully controlled. (A message notifies you when IMAGE/SQL commands containing maintenance words or passwords have been logged.)

About IMAGE/SQL Security

IMAGE/SQL enforces TurboIMAGE/XL database security. That is, SQL users can access only the data defined for them in the TurboIMAGE/XL database schema.

To accomplish this, during an attach, only the DBC is defined as an SQL user. This user has access to all the mapped tables in the database. The DBC must explicitly add all other IMAGE/SQL users by associating each user class or password with a DBEUser_ID. A view is then created for each mapped table to which the DBEUser_ID has access. This view is based on information in the TurboIMAGE/XL root file and permits user classes to see the data defined for them in the TurboIMAGE/XL schema. Users, including user class 0, must know the names of the views to which they have access.

IMAGE/SQL utility security can be modified only by someone who is both the DBC and a DBA of the respective database management systems.

Controlling IMAGE/SQL User Access

IMAGE/SQL takes the following steps to control users' access to TurboIMAGE/XL data (see Figure 3-4):

- When an SQL user is added, IMAGE/SQL creates an SQL authorization group. The name of this group is based on the user class or the user class of the password named in the ADD USER command. The naming convention for this group is *OwnerName_UserClass#*. The new DBEUserID (*User@Account*) is then added to this group.

  One view is then created for each data set the user class is allowed to access. The naming convention for these views is *OwnerName.MappedTableName_V UserClass#*.

  If data is not password-protected, user class 0 is assumed and views of this data are created for user class 0.

- When an SQL user is deleted, the DBEUserID is removed from the SQL group associated with the TurboIMAGE/XL user class. Note that the group itself and the corresponding views remain in the DBEnvironment because other user-created views may be based on these views.

- When the database is detached, all views based on mapped tables, including user-created views, are dropped.

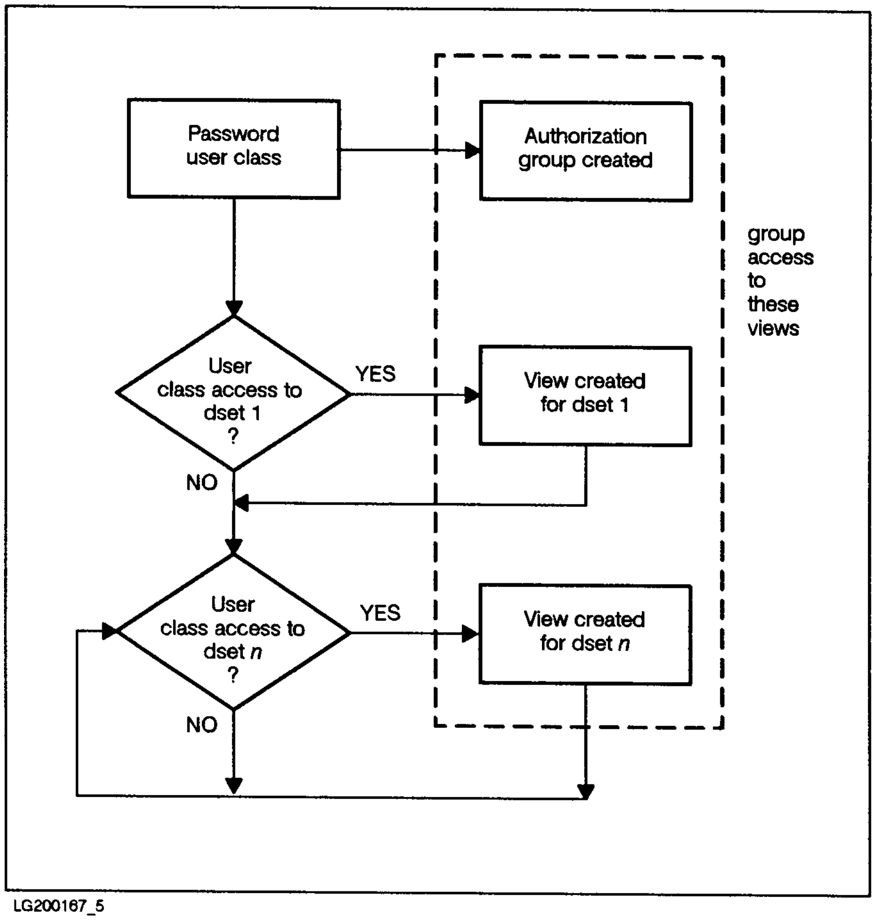

Password user class
Authorization group created
group access to these views
User class access to dset 1 ?
YES
NO
View created for dset 1
User class access to dset n ?
YES
NO
View created for dset n
LG200167_5

Figure 3-4. IMAGE/SQL Security Mapping

IMAGE/SQL Data Type Mapping

IMAGE/SQL maps all TurboIMAGE/XL data types to the closest equivalent SQL data types. Sometimes completely compatible choices are not available, or more than one viable alternative exists. In these cases, IMAGE/SQL chooses default data types for you, but also provides alternative data type mapping that you can select if it more closely meets your particular needs.

Note that when alternative data type mapping is selected, all user-created views containing the mapped data type are dropped. Therefore, it is advisable to perform alternative data type mapping before users have had the opportunity to create views.

For specific information about IMAGE/SQL default data type mapping and alternative choices, refer to Table 2-6 in Chapter 2 (Task 4).

At Run Time

At run time, SQL turns all mapped table queries over to IMAGE/SQL. Using the mapping information in the ATCINFO file, IMAGE/SQL makes the appropriate TurboIMAGE/XL calls, retrieves the data from the TurboIMAGE/XL database, and returns the data to SQL in the correct SQL format (see Figure 3-5.)

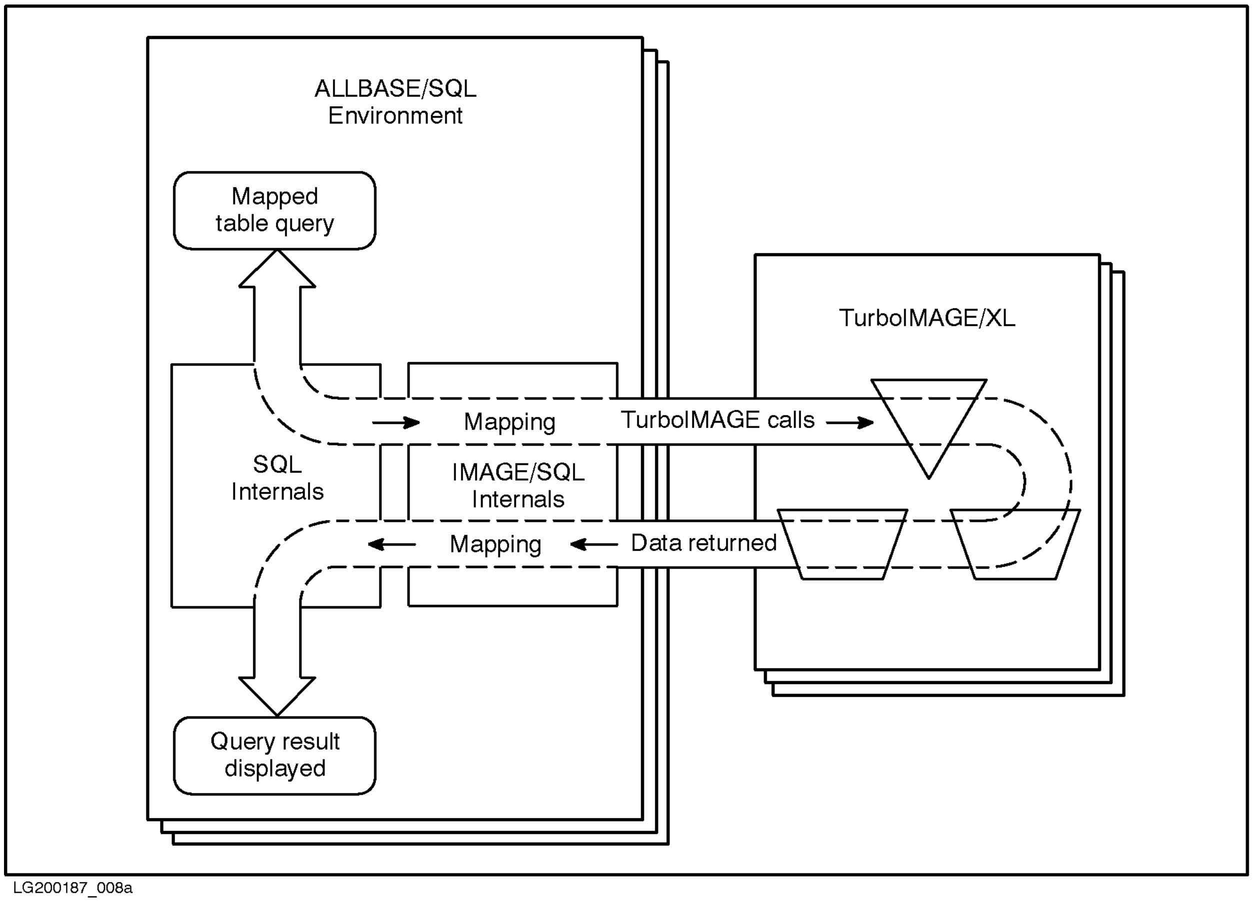


Figure 3-5. IMAGE/SQL at Run Time

Note that the data is retrieved from the TurboIMAGE/XL database. Only the mapped table definitions actually reside in the DBEnvironment. The ALLBASE/SQL Optimizer decides which indices, if any, to use and the proper order of operation to ensure that the most efficient path is used. Data is retrieved more efficiently when a mapped column represents a TurboIMAGE/XL search item, key item, or an item which has a B-Tree index (explicit or implicit) or a third-party index.

Accessing Mapped Tables

IMAGE/SQL provides transparent access to TurboIMAGE/XL data. However, IMAGE/SQL users should keep in mind the considerations described in this section when accessing TurboIMAGE/XL data.

Performance Considerations

The following factors can affect IMAGE/SQL performance:

Up-to-Date Information

For better performance, it is important to have accurate statistics on the tables. Use the SQL UPDATE STATISTICS statement to ensure that this information is up-to-date. Mapped table statistics should be updated immediately after the database is attached and again when the number of data entries in mapped tables has changed significantly. Refer to the *ALLBASE/SQL Database Administration Guide* for information about how to use this statement.

Items in Queries

When a mapped column represents a TurboIMAGE/XL key item, search item, or the item which has a third-party index or an explicit or implicit B-Tree index, IMAGE/SQL uses this item to retrieve the data more efficiently whenever possible. (An explicit B-Tree index is on a master's key item, while an implicit B-Tree index is on a detail's search item whose related key item of the master has a B-Tree index.) Because of this, using such a mapped column in the WHERE clause of an SQL SELECT statement can improve performance in certain circumstances. Task 15, "Selecting TurboIMAGE/XL Data with SQL," in Chapter 2 describes these circumstances and gives examples of their use.

IMAGE/SQL Utility Commands

This chapter contains detailed information about the commands for the IMAGE/SQL utility, IMAGESQL.

The following is included for each command:

- Command syntax
- Explanation of the command's parameters
- Prerequisites needed to execute the command
- Discussion of the command's functionality
- Example(s)

In all examples, the TurboIMAGE/XL database shown is SALES and the DBEnvironment is PartsDBE. Refer to Figure 2-1 in Chapter 2 to see a summary of the prerequisites necessary to execute IMAGE/SQL utility commands. The prerequisites are also listed for each command in this chapter. For example, you must log on as DBA to use the DETACH command, while either the DBA or the DBC can use the ADD USER command. Run the IMAGE/SQL utility by issuing the command:

```
:RUN IMAGESQL.PUB.SYS
```

ADD USER

Adds an IMAGE/SQL user.

Syntax

```
AD[D] USER { User@Account } WITH { PASS=Password  }
           { User.Account }      { CLASS=Classnum }
[,MODE=ModeNumber]
```

Parameters

User@Account	is the name used to identify the new user to IMAGE/SQL. This name, referred to in SQL as the DBEUserID, is made up of an MPE/iX user and account name, connected with the period or @ symbol. It must contain valid logon syntax. If the account is omitted, it will default to the logon account.
Password	is a password in the TurboIMAGE/XL schema. The new IMAGE/SQL user has the same access to TurboIMAGE/XL data as the user class associated with this password. The password is case-sensitive.
Classnum	is the user class number.
ModeNumber	is the DBOPEN mode with which the TurboIMAGE/XL database is opened for this user. Valid DBOPEN modes supported by IMAGE/SQL are 1 through 8. If this parameter is omitted, the mode defaults to 5. Refer to the *TurboIMAGE/XL Database Management System Reference Manual* for information about these DBOPEN modes.

Prerequisites

- SET SQLDBE issued.
- SET TURBODB issued.
- DBC status.
- DBA authority.
- Database attached.

Description

Use the ADD USER command to add a new IMAGE/SQL user. When a user is added, an SQL authorization group is created that corresponds to the user class or the user class of the password given in the command. The new DBEUserID (*User@Account*) is then added to this group. The group is named according to the following conventions:

OwnerName_UserClassNumber

By default, *OwnerName* is the name of the database (unqualified by its group and account), but a different *OwnerName* can be specified at attach time. For example, for a database owned by MKTG, the group created for user class 11 is named MKTG_11.

To enforce TurboIMAGE/XL security, one view is created for each data set the user class is allowed to access. The view is named according to the following conventions:

OwnerName.*MappedTableName*_V*UserClassNumber*

For example, for a database owned by MKTG, the view created for the mapped table ACCOUNTS is MKTG.ACCOUNTS_V11 for user class 11. Each view contains only those mapped columns that correspond to the fields in the data set the user class can read. The corresponding authorization group is then granted access to these views.

Example

In the following example, user RYAN.ATC is being added as an IMAGE/SQL user. The database will be opened in DBOPEN mode 1. His access to data is the same as that allowed to the password "manager". Because the password "manager" is associated with user class 18, this number appears at the end of each view name.

```
>>ADD USER  RYAN.ATC WITH PASS=manager, MODE=1
Warning: command containing a password has been logged (ATCWARN 32069).
ALLBASE/SQL group SALES_18 created.
View SALES.DATE_MASTER_V18 created.
View SALES.CUSTOMER_V18 created.
View SALES.PRODUCT_V18 created.
View SALES.VENDOR_V18 created.
View SALES.INVENTORY_V18 created.
View SALES.SALES_V18 created.
>>
```

In the above example, using CLASS=18 instead of PASS=manager will have the same result.

ATTACH

Attaches a TurboIMAGE/XL database to an SQL DBEnvironment.

Syntax

```
AT[TACH] [WITH OWNER=OwnerName]
```

Parameters

OwnerName	specifies an owner for all SQL objects that need to be created for the attached TurboIMAGE/XL database. This name can be up to 17 bytes in length and can be made up of any combination of letters (A to Z), decimal digits (0 to 9), $, #, @, or _ (underscore). However, the first character cannot be a decimal digit or an underscore. Lowercase letters are automatically converted to uppercase letters. Note that group and account names are not included in the *OwnerName* parameter. If this parameter is omitted, the owner name defaults to the name of the TurboIMAGE/XL database defined in the most recent SET TURBODB command.

Prerequisites

- SET SQLDBE issued.
- SET TURBODB issued.
- DBA authority.
- Database detached.

Description

Use the ATTACH command to attach a TurboIMAGE/XL database to a DBEnvironment. This command can only be used after the database name and the DBEnvironment name have been specified with SET commands.

You need exclusive access to the DBEnvironment when using the ATTACH command, because it is required for the first ATTACH command. Use the ISQL SELECT command to find out if it is being accessed by users.

```
:RUN ISQL.PUB.SYS
>isql=> SELECT * FROM SYSTEM.USER;
>isql=> exit;
```

When a database is attached to a DBEnvironment, only the DBC is defined as an IMAGE/SQL user and default data type mapping is performed. Once attached, IMAGE/SQL utility commands can be used to update this default information. Refer to the ADD USER, UPDATE USER, UPDATE TYPE, and SPLIT commands for more information.

A database already attached to a specific DBEnvironment cannot be reattached. If you attempt to do this, an error message is issued. If you want to reattach it, you must perform a DETACH first and then ATTACH it.

To attach databases with the same database name but in different groups and accounts to the same DBEnvironment, you must use the *OwnerName* parameter to specify a different owner name for all but the first such database you attach.

If the DBEnvironment does not exist, IMAGE/SQL displays this message:

```
   DBE does not exist, do you want to create one? [Y/N] :
```

If you reply 'Y', a DBEnvironment and DBE files are automatically created for you. The files created are:

File Created	File Name
DBEConFile	*DBEnvironmentName*
DBEFile	*DBEnvironmentName*`FL`
LOG file	*DBEnvironmentName*`LG`
ATCINFO file	*DBEnvironmentName*`CR`

For example, if you issue the command

```
   SET SQLDBE MYDBE
```

and the DBEnvironment MYDBE does not exist, these files are created: MYDBE, MYDBEFL, MYDBELG, and MYDBECR.

The size of DBEFile is 5000 pages and LOG file is 10000 pages.

`ATTACH` triggers entering definitions for hash indices on all TurboIMAGE/XL master keys and detail search items in the system catalog of the DBEnvironment specified by the `SET SQLDBE` statement. All master keys, except P and Z data types, are entered as having unique hash indices. All detail search items, as well as master P and Z key types, are entered as having non-unique hash indices. With hash indices, performance gain can be observed only when the mapped column of the key or search item in an SQL statement employs an equality operator. That is, the SQL Optimizer will only derive an index scan on this mapped column if the operator used with it is "=". For other operators (such as > or <) used with this mapped column, the Optimizer will not choose the hash index scan on this mapped column.

The name of the hash index on the key item is derived by suffixing the mapped column name of the key item with '_A1' for the automatic master and '_M1' for the manual master. For the detail data set, '_Dn' is appended to the mapped column name of the search item where n is the path number. The definitions for hash indices can be seen in the views, SYSTEM.IMAGEKEY and CATALOG.IMAGEKEY, of the system catalog of the

DBEnvironment. For an example, the SQL statement, Select * from SYSTEM.IMAGEKEY, after connecting to the DBEnvironment will display all hash indices.

If B-Tree indices are created on the key items of the selected master data sets using DBUTIL or DBSCHEMA, IMAGE/SQL enters definitions for the B-Tree indices in the system catalog of the DBEnvironment. Although the B-Tree index can only be created on the key item, the definitions for B-Tree indices on all of its related search items of the detail data sets are also entered in the SQL catalog. The B-Tree index on the key item of the master set is perceived as an EXPLICIT index and that on the related search item as IMPLICIT (virtual). For an example, if a B-Tree index is created on the key item which has paths to 16 detail data sets, a definition for an index for all 16 data sets will be entered. That is, this will result into 17 (1 for master and 16 for detail data sets) definitions. The index on the key item of the master, except for P and Z types, is entered as 'unique' index. Other definitions, by default, are non-unique. The index name for the key items is derived by suffixing its mapped column name with '_B1'. For the related search item, its mapped column is suffixed with '_Vn' where n is the path number. With B-Tree indices, you can use operators such as '<=', and '> ' which are permissible in ALLBASE/SQL. The B-Tree indices can be seen in the views, SYSTEM.INDEX and CATALOG.INDEX, of the DBEnvironment.

If the key item or its related search item is split using the SPLIT command of IMAGESQL, definitions for both hash index and B-Tree index on the split item will not be entered. For an example, a key item on which a B-Tree index exists, is split, but its related search item in the detail set is not. Definitions for unique hash index and unique B-Tree index on this split key item will not be entered. However, definitions for non-unique hash index as well as non-unique B-Tree index on the related search item will be entered.

If the database is enabled for third-party indices (TPI), ATTACH enters definitions for all TPIs, excluding keyword indices and the ones for which the third-party does not provide information. The TPIs are entered as unique or non-unique based on the index configuration and information provided by the third-party for this index.

While keys, search items, and B-Tree indices are registered in the specified DBEnvironment, third-party indices are registered in all attached DBEnvironments.

Multiple index definitions on the same column can coexist and the SQL optimizer derives the optimal access plan based on the statistics present in the system catalog. In other words, the key or search item of the set can have a maximum of three index definitions. One will be a hash index definition entered automatically at ATTACH time, another can be a B-Tree index definition, and the third can be a third-party index definition. It is recommended that both B-Tree index and third-party index be not created on the same item as it will have an unnecessary impact on the performance (Optimizer calculates cost for each index).

The Optimizer decides which index to use and the proper order of operations to ensure that the most efficient path is used.

Example

In the following example, SALES is attached to PARTSDBE. The accompanying message summarizes the mapping that took place during the attach.

```
>>SET TURBODB SALES
>>SET SQLDBE PARTSDBE
>>ATTACH
Split 1 compound source field(s) (ATCWARN 32063).
Mapped 15 source table/source field name(s) (ATCWARN 32062).
Mapped 1 incompatible source type(s) (ATCWARN 32061).
>>
```

To see the specific mapping for each data set and field, use the DISPLAY MAP command. In the following example, the display notes that the data type mapping performed for the mapped column CREDIT_RATING is imprecise. Also noted is the splitting of the compound source field OTHER_VENDORS into three mapped columns.

```
>>DISPLAY MAP

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC
Owner Name       : SALES

MAPPED(SOURCE)   SOURCE             MAPPED       SOURCE     MAPPED
   TABLE         FIELD              COLUMN        TYPE       TYPE    NOTES
------------ ---------------- ------------------ ------- ----------- -----

DATE_MASTER (DATE-MASTER)
             DATE             DATE               X6      CHAR(6)

CUSTOMER (CUSTOMER)
             CUSTOMER#        CUSTOMER#          J2      INTEGER
             LAST-NAME        LAST_NAME          X16     CHAR(16)
             FIRST-NAME       FIRST_NAME         X10     CHAR(10)
             INITIAL          INITIAL            U2      CHAR(2)
             STREET           STREET             X26     CHAR(26)
             CITY             CITY               X12     CHAR(12)
             STATE            STATE              X2      CHAR(2)
             ZIP              ZIP                X6      CHAR(6)
             CREDIT-RATING    CREDIT_RATING      R2      FLOAT       I

PRODUCT (PRODUCT)
             PRODUCT#         PRODUCT#           U8      CHAR(8)
             PRODUCT-DESCRIPT PRODUCT-DESCRIPT   X20     CHAR(20)


VENDOR (VENDOR)
             VENDOR           VENDOR             X16     CHAR(16)
             STREET           STREET             X26     CHAR(26)
             CITY             CITY               X12     CHAR(12)
             STATE            STATE              X2      CHAR(2)

INVENTORY (INVENTORY)
             PRODUCT#         PRODUCT#           U8      CHAR(8)
             ON-HAND-QTY      ON_HAND_QTY        J2      INTEGER
             VENDOR           VENDOR             X16     CHAR(16)
             OTHER-VENDORS    OTHER_VENDORS_1    X16     CHAR(16)     S
             OTHER-VENDORS    OTHER_VENDORS_2    X16     CHAR(16)     S
             OTHER-VENDORS    OTHER_VENDORS_3    X16     CHAR(16)     S
             UNIT-COST        UNIT_COST          P8      DECIMAL(7,0)
             LAST-SHIP-DATE   LAST_SHIP_DATE     X6      CHAR(6)
             LOCATION-BIN     LOCATION_BIN       Z2      DECIMAL(2,0)
             PART-INFO        PART_INFO          X60     CHAR(60)

SALES (SALES)
             CUSTOMER#        CUSTOMER#          J2      INTEGER
             PRODUCT#         PRODUCT#           U8      CHAR(8)
             QUANTITY         QUANTITY           I1      SMALLINT
             PRICE            PRICE              J2      INTEGER
             TAX              TAX                J2      INTEGER
             TOTAL            TOTAL              J2      INTEGER
             PURCHASED-DATE   PURCHASED_DATE     X6      CHAR(6)
             DELIVERED-DATE   DELIVERED_DATE     X6      CHAR(6)

NOTES:
  I: Imprecise(float)/Incompatible(others) mapping between source and
     mapped data types
  S: Source field has been split
>>
```

DELETE USER

Deletes an IMAGE/SQL user.

Syntax

$$\texttt{DEL[ETE] USER} \left\{ \begin{array}{l} \textit{User@Account} \\ \textit{User.Account} \end{array} \right\}$$

Parameters

User@Account	is the name used to identify the IMAGE/SQL user to SQL. This name, referred to as the DBEUserID, is made up of an MPE/iX user and account name, connected with the period or @ symbol. This user and account must be a valid existing logon.

Prerequisites

- SET SQLDBE issued.
- SET TURBODB issued.
- DBC status.
- DBA authority.
- Database attached.

Description

Use the DELETE USER command to delete an IMAGE/SQL user from a DBEnvironment. When a user is deleted, the DBEUserID (*User@Account* or *User.Account*) is removed from the SQL group associated with the TurboIMAGE/XL user class. Note that the group itself and the corresponding view(s) remain in the DBEnvironment because other user-created views may be based on views created by IMAGE/SQL.

Example

In the example below, once the DELETE USER command is issued, user RYAN.ATC can no longer access the SALES database with IMAGE/SQL.

```
>>DELETE USER  RYAN.ATC
>>
```

DETACH

Detaches a TurboIMAGE/XL database from a DBEnvironment.

Syntax

```
DET[ACH]
```

Command Prerequisites

- SET SQLDBE issued.
- SET TURBODB issued.
- DBA authority.
- Database attached.

Description

Use the DETACH command to detach a TurboIMAGE/XL database from a DBEnvironment. Obtain exclusive access to the DBEnvironment before using the DETACH command. Use the ISQL SELECT command to find out if it is being accessed. (See the ATTACH command for an example of SELECT.)

SET TURBODB and SET SQLDBE commands must be issued before the detach can take place. The database named in the most recent SET TURBODB is detached from the DBEnvironment named in the most recent SET SQLDBE command.

You must detach the database before performing any restructuring tasks and reattach (using the ATTACH command) when the restructuring is complete. You do not need to detach the database when changing the capacity of data sets.

When a database is detached, all mapping information about it is removed from the ATCINFO file (*DBEnvironmentName*CR). If it is the only database attached to the DBEnvironment, the ATCINFO file itself is purged. If the TurboIMAGE/XL database is no longer attached to any DBEnvironment, the *DBName*TC file is also purged.

In addition, all mapped table definitions are removed from the DBEnvironment system catalog. All IMAGE/SQL views and user-created views based on IMAGE/SQL tables and index definitions are dropped. Note that DBUTIL can be used to purge and detach a database that is attached to a DBEnvironment.

`DETACH` only affects the DBEnvironment named in the `SET SQLDBE` statement. Information about tables and indices is removed from the specified DBEnvironment.

Example

In this example, SALES is detached from PartsDBE. When this occurs, all mapping information about SALES is removed from the ATCINFO file. If SALES is the only database attached to PartsDBE, the ATCINFO file itself is purged. If SALES is no longer attached to any DBEnvironment, the SALESTC file is also purged.

```
:RUN IMAGESQL.PUB.SYS
>>SET SQLDBE PARTSDBE
>>SET TURBODB SALES
>>DETACH
```

DISPLAY MAP

Displays the current data type mapping information for a specified TurboIMAGE/XL database.

Syntax

```
DI[SPLAY] MAP [FOR TurboDB [,MAINT=TurboMaintWord] ]
[MappedTable[.MappedCol]] [,...]
```

Parameters

TurboDB	is the name of the TurboIMAGE/XL database whose data type mapping information you want to display. This name can be a fully qualified group and account name. If this parameter is omitted, the name of the database defined in the most recent SET TURBODB command is used.
TurboMaintWord	is the maintenance word for the TurboIMAGE/XL database. This parameter can be omitted if you are the database creator (DBC).
MappedTable	is the name of a mapped table whose data type mapping information you want to display. If this parameter is omitted, information about all mapped tables is displayed.
MappedCol	is the name of a specific mapped column in the table whose data type mapping information you want to display. If this parameter is omitted, information about all columns in the table is displayed.

Prerequisites

- SET SQLDBE issued.
- SET TURBODB issued or TurboIMAGE/XL database (and maintenance word, if not DBC) specified as part of the command.
- DBA authority.
- Database attached.

Description

Use the DISPLAY MAP command to display name and data type mapping information for a TurboIMAGE/XL database attached to an SQL DBEnvironment. Information is displayed about the DBEnvironment named in the most recent SET SQLDBE command.

If the database is attached to more than one DBEnvironment, issue additional SET SQLDBE commands to see the data type mapping for each DBEnvironment to which it is attached.

In the display generated by this command:

- An I in the NOTES section next to a particular mapping indicates that the default data type mapping is inexact or imprecise.
- An S in the NOTES section next to a particular mapping indicates that the mapped column has been split. This occurs by default when the source field is a compound data item. The SPLIT

command may also have been used to explicitly split a mapped column into smaller mapped columns. In this case, a U also appears in the NOTES section.

- A U in the NOTES section next to a particular mapping indicates that the default data type mapping has been updated with the UPDATE TYPE command.

Refer to the SPLIT and UPDATE TYPE commands for information about how to select alternative data type mapping.

Example

In the following example, mapping information for the mapped table INVENTORY is displayed. The compound source field OTHER–VENDORS has been split into three mapped columns: OTHER_VENDORS_1, OTHER_VENDORS_2, and OTHER_VENDORS_3. Note that the source data set name, in this case INVENTORY, is shown in parentheses after the name of the mapped table.

```
>>DISPLAY MAP INVENTORY

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC
Owner Name       : SALES

MAPPED(SOURCE)   SOURCE            MAPPED       SOURCE     MAPPED
   TABLE         FIELD             COLUMN        TYPE       TYPE    NOTES
------------ ---------------- ------------------ ------- ----------- -----



INVENTORY (INVENTORY)
             STOCK#           STOCK#             U8      CHAR(8)
             ONHANDQTY        ONHANDQTY          J2      INTEGER
             SUPPLIER         SUPPLIER           X16     CHAR(16)
             OTHER-VENDORS    OTHER_VENDORS_1    X16     CHAR(16)    S
             OTHER-VENDORS    OTHER_VENDORS_2    X16     CHAR(16)    S
             OTHER-VENDORS    OTHER_VENDORS_3    X16     CHAR(16)    S
             UNIT-COST        UNIT_COST          P8      DECIMAL(7,0)
             LASTSHIPDATE     LASTSHIPDATE       X6      CHAR(6)
             BINNUM           BINNUM             Z2      DECIMAL(2,0)
             PART-INFO        PART_INFO          X60     CHAR(60)

NOTES:
  S: Source field has been split
>>
```

DISPLAY OPTIONS

Displays the options in effect for the current IMAGE/SQL utility session.

Syntax

```
DI[SPLAY] OPTIONS
```

Prerequisites

None.

Description

Use the DISPLAY OPTIONS command to display the options in effect for the current IMAGE/SQL utility session. The following information is displayed:

- The database named in the last SET TURBODB command.
- The DBEnvironment named in the last SET SQLDBE command.
- The condition specified in the last LOG command (on or off).
- The condition specified in the last ECHO command (on or off).
- The name of the IMAGE/SQL utility log file (only if logging is on).
- The attached/detached status of the database named in the SET TURBODB command.

Example

In this example, SALES.SERED.ATC is the database named in the most recent SET TURBODB command and PARTSDBE.SERED.ATC is the DBEnvironment named in the most recent SET SQLDBE command. Echo is on, logging is on, and the database is attached to the DBEnvironment.

```
>>DISPLAY OPTIONS
Current base         : SALES.SERED.ATC
Current SQLDBE       : PARTSDBE.SERED.ATC
Echo                 : ON
Command Logging      : ON
Log File             : ATCLOG.SERED.ATC

*** Database is attached.
>>
```

DISPLAY SQLDBES

Displays all the DBEnvironments currently attached to a TurboIMAGE/XL database.

Syntax

```
DI[SPLAY] SQLDBE[S] [FOR TurboDB
[,MAINT=TurboMaintWord]]
```

Parameters

TurboDB	is the name of the TurboIMAGE/XL database whose associated DBEnvironments you want to display. If this parameter is omitted, the database must be previously specified with the SET TURBODB command or else an error message is issued.
TurboMaintWord	is the maintenance word for the TurboIMAGE/XL database. This parameter can be omitted if you are the database creator (DBC).

Prerequisites

SET TURBODB issued or TurboIMAGE/XL database (and maintenance word, if not DBC) specified as part of the command.

Description

Use the DISPLAY SQLDBES command to display the DBEnvironments to which the TurboIMAGE/XL database is currently attached.

Example

In the following example, SALES.SERED.ATC is only associated with PARTSDBE.SERED.ATC.

```
>>DISPLAY SQLDBES FOR SALES.SERED.ATC

ATTACHED BASES : SALES.SERED.ATC

DBENVIRONMENT           OWNER
-------------           -----
PARTSDBE.SERED.ATC      SALES
>>
```

DISPLAY TURBODBS

Displays all TurboIMAGE/XL databases currently attached to a DBEnvironment and, optionally, their associated mapped tables.

Syntax

DI[SPLAY] TURBODB[S] [TABLES] [FOR *DBEnvironment* [,MAINT=*DBE_ MaintWord*]]

Parameters

TABLES	is an option that displays all mapped tables in each TurboIMAGE/XL database attached to the DBEnvironment. If this parameter is omitted, no table information is displayed.
DBEnvironment	is the name of the DBEnvironment whose attached TurboIMAGE/XL databases you want to see. If this parameter is omitted, the DBEnvironment must have been previously specified with the SET SQLDBE command or else an error message is issued.
DBE_MaintWord	is the maintenance word for the DBEnvironment. This parameter can be omitted if you are the DBECreator.

Prerequisites

- SET SQLDBE issued or DBEnvironment (and maintenance word, if not DBECreator) specified as part of the command.

Description

Use the DISPLAY TURBODBS command to display information about all of the TurboIMAGE/XL databases attached to a DBEnvironment. Optionally, all tables in each TurboIMAGE/XL database are also listed.

The DISPLAY BASE command is synonymous with the DISPLAY TURBODBS command and has the same parameters and prerequisites.

Example

In the following example, the databases currently attached to the PARTSDBE.SERED.ATC DBEnvironment are listed.

```
>>DISPLAY TURBODBS FOR PARTSDBE.SERED.ATC

DBEnvironment     : PARTSDBE.SERED.ATC

BASE FILE               OWNER          BASE TYPE
---------               -----          ---------

SALES1.SERED.ATC        SALES1         TurboIMAGE
SALES2.SERED.ATC        SALES2         TurboIMAGE
DB1.SERED.ATC           SERED          TurboIMAGE
DB2.SERED.ATC           DB2            TurboIMAGE

Total Databases : 4

>>
```

DISPLAY USERS

Displays current user information for a specific TurboIMAGE/XL database.

Syntax

DI[SPLAY] USER[S] [FOR *TurboDB*] [USER= $\left\{ \begin{array}{l} \textit{User@Account} \\ \textit{User.Account} \end{array} \right\}$ [,... .]]

Parameters

TurboDB	is the name of the TurboIMAGE/XL database whose user information you want to display. If this parameter is omitted, the name of the database defined in the most recent SET TURBODB command is used.
User@Account	is the name used to identify a particular user in the DBEnvironment. This name, referred to as the DBEUserID, is made up of the user's MPE/iX user and account names, connected with the period or @ symbol. If this parameter is omitted, information about all currently defined users is displayed.

Prerequisites

- SET SQLDBE issued.
- SET TURBODB issued or TurboIMAGE/XL database specified as part of the command.
- DBC status.
- Database attached.

Description

Use the DISPLAY USERS command to display information about users of the TurboIMAGE/XL database named in the last SET TURBODB command. For the named TurboIMAGE/XL database, the DBOPEN mode, the passwords, and classes are displayed for all IMAGE/SQL users or for those users specified by the *User@Account* parameter when viewed by DBC.

A user who is not the DBC, can view user information without displaying passwords.

To change the DBOPEN mode or the user class for a specified user, refer to the UPDATE USER command. To add or delete users, refer to the ADD USER and DELETE USER commands.

Example

In the following example, information is displayed about user NANCY@ATC.

```
>>DISPLAY USER USER=NANCY@ATC

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC
Owner Name       : SALES

USER LOGON         DBOPEN MODE  USER PASSWORD  USER CLASS
----------         -----------  -----------    ----------

NANCY.ATC          1            ;              64

>>
```

ECHO

Determines if commands and comments are echoed during the execution of a command file.

Syntax

$$\texttt{EC[HO]}\left\{\begin{array}{l}\texttt{ON}\\\texttt{OFF}\end{array}\right\}$$

Parameters

`ON`	turns echo on. This is the default.
`OFF`	turns echo off.

Prerequisites

None.

Description

Use the ECHO command to turn the IMAGE/SQL utility echo on or off. In session mode, when echo is on, all commands issued from a command file and any accompanying comments are displayed at the terminal. Regardless of the setting of this command, however, normal command output and error messages are displayed at the terminal.

When the IMAGE/SQL utility is run, echo is initially on. To see the current echo status, use the DISPLAY OPTIONS command.

Example

In this example, the DISPLAY OPTIONS command first shows that echo is on. The ECHO OFF command is then issued. The DISPLAY OPTIONS command confirms that echo is now off.

```
>>DISPLAY OPTIONS
Current TurboDB     : SALES.SERED.ATC
Current SQLDBE      : PARTSDBE.SERED.ATC
Echo                : ON
Command Logging     : ON
Log File            : ATCLOG.SERED.ATC

*** Database is attached.
>>ECHO OFF
>>DISPLAY OPTIONS
Current TurboDB     : SALES.SERED.ATC
Current SQLDBE      : PARTSDBE.SERED.ATC
Echo                : OFF
Command Logging     : ON
Log File            : ATCLOG.SERED.ATC

*** Database is attached.
>>
```

EXIT

Stops execution of the IMAGE/SQL utility program.

Syntax

```
EX[IT]
```

Prerequisites

None.

Description

Use the EXIT command to leave the IMAGE/SQL utility program. This command provides the same functionality as the QUIT command.

Example

In the following example, the IMAGE/SQL utility stops execution and returns you to the MPE/iX prompt.

```
>> EXIT

END OF PROGRAM
:
```

HELP

Provides information about IMAGE/SQL utility commands.

Syntax

$\left\{\begin{array}{l}\texttt{H[ELP]}\\ \texttt{?}\end{array}\right\}$ [*CommandName* [*KeyWord*]]

Parameters

Command-Name	is the name of the IMAGE/SQL utility command about which you want information. The abbreviated form of the command can be used. If this parameter is omitted, syntax information about all IMAGE/SQL utility commands is displayed.
KeyWord	is a word that further defines some commands. For example, in the DISPLAY MAP command, MAP is a keyword. If this parameter is omitted, help is displayed for all keywords associated with a particular command name.

Prerequisites

None.

Description

Use the HELP command to display information about IMAGE/SQL utility commands.

Example 1

In the following example, because no IMAGE/SQL utility command name has been specified as a parameter of the HELP command, the syntax for all IMAGE/SQL utility commands is displayed.

```
>>HELP
The following commands are available:

AD[D] USER              AT[TACH]                DEL[ETE] USER
DET[ACH]                DI[SPLAY] MAP           DI[SPLAY] OPTIONS
DI[SPLAY] SQLDBE[S]     DI[SPLAY] TURBODB[S]    DI[SPLAY] USER[S]
EC[HO]                  EX[IT]                  H[ELP]
L[OG]                   Q[UIT]                  RECOVER
REDO                    SE[T] SQLDBE            SE[T] TURBODB
SP[LIT]                 U[PDATE] TYPE           U[PDATE] USER
X[EQ]

For help on a particular command, type : 'HELP CommandName [KeyWord]'
>>
```

Example 2

In the following example, the HELP command is qualified with the IMAGE/SQL utility command DISPLAY and the keyword SQLDBES. The syntax and an example of this command/keyword are displayed.

```
>>HELP DISPLAY SQLDBES
DI[SPLAY] SQLDBE[S]
----------------
   Displays all the DBEnvironments associated with a specific TurboIMAGE/XL
   database.

SYNTAX
   DI[SPLAY] SQLDBE[S] [FOR TurboDBName  [,MAINT = TurboMaintWord]]


EXAMPLE
   DISPLAY SQLDBES FOR MYTURBODB
>>
```

LOG

Turns IMAGE/SQL command logging on or off.

Syntax

```
L[OG] { ON  }
      { OFF }
```

Parameters

`ON` turns command logging on. This is the default.

`OFF` turns command logging off.

Prerequisites

None.

Description

Use the LOG command to turn IMAGE/SQL utility command logging on or off. When logging is on, all commands entered are logged to the ATCLOG file.

When the IMAGE/SQL utility is run, logging is initially on. The temporary file ATCLOG is created (if it does not already exist) and opened in the user's logon group and account. A file equation may be set for this file (:FILE ATCLOG=LOG1). If you want to save the temporary file, either save the file before ending your current MPE/iX session or specify ;SAVE as a part of a file equation. If the log file already exists, log records are appended to it.

To see the current logging status, use the DISPLAY OPTIONS command.

When you issue an MPE/iX command from within IMAGE/SQL, the command is written to the log file and the log file is closed. This means that you can save this file, edit this file, or set a file equation for this file from within IMAGE/SQL. Refer to Chapter 2, "Logging IMAGE/SQL Utility Commands: Task 13," for more information. The log file is reopened after the execution of the MPE/iX command.

Example

In the following example, logging is initially on. The LOG OFF command is then issued. The DISPLAY OPTIONS command then confirms that logging is off.

```
>>DISPLAY OPTIONS
Current TurboDB      : SALES.SERED.ATC
Current SQLDBE       : PARTSDBE.SERED.ATC
Echo                 : ON
Command Logging      : ON
Current Log File     : ATCLOG.SERED.ATC
>>LOG OFF
>>DISPLAY OPTIONS
Current TurboDB      : SALES.SERED.ATC
Current SQLDBE       : PARTSDBE.SERED.ATC
Echo                 : ON
Command Logging      : OFF
>>
```

QUIT

Stops the execution of the IMAGE/SQL utility program.

Syntax

```
Q[UIT]
```

Prerequisites

None.

Description

Use the QUIT command to leave the IMAGE/SQL utility program. This command provides the same functionality as the EXIT command.

Example

In the following example, the IMAGE/SQL utility stops execution and returns you to the MPE/iX prompt.

```
>>QUIT

END OF PROGRAM
:
```

RECOVER

Performs maintenance on the ATCINFO file.

Syntax

```
RECOVER
```

Prerequisites

None.

Description

Use the RECOVER command to perform maintenance on the ATCINFO file. The default name of the ATCINFO file is *DBEnvironmentName*CR with up to six characters of the *DBEnvironmentName*. Maintenance for the ATCINFO file may be necessary in either of the following situations:

- If a crash occurs while the ATCINFO file is being modified. This may be the case if a crash takes place when the IMAGE/SQL administrator is in the midst of an IMAGE/SQL utility command that updates the ATCINFO file. When this occurs, the RECOVER command can be used to reconstruct the ATCINFO file.
- If the ATCINFO file contains too much free space. This may be the case if multiple databases are attached and then some are detached from the DBEnvironment. When this occurs, the RECOVER command can be used to compact the data in the ATCINFO file and return the free space to the file system.

Example

In the following example, the ATCINFO file associated with the PartsDBE DBEnvironment (PARTSCR) is recovered.

```
>>SET SQLDBE PartsDBE
>>RECOVER
Checking physical file consistency and recovering free space.
Deleting unused mapped table entries.
Checking external cross references.
>>
```

REDO

Allows a user to correct or modify the last IMAGE/SQL utility command.

Syntax

```
REDO
```

Prerequisites

None.

Description

Use this command to display the last IMAGE/SQL utility command entered. Use the associated subcommands to correct or change the displayed command. The REDO command applies only to the last command entered and is available only in interactive mode. It is ignored with a warning in batch mode.

Subcommands

`D`	deletes the character above the cursor. If D is repeated, each character above each D is deleted.
`I`	inserts one or more characters immediately preceding the character above the cursor. The D and I subcommands can be used in conjunction to delete characters and then insert new characters.
`R`	replaces the characters above the cursor with new characters. If one character is entered, the character above the cursor is replaced; if two characters are entered, two characters (the character above the cursor and the character to the right) are replaced; and so forth for additional characters. R is the default subcommand and is only required when the character to be replaced is a D, I, R, or U.
`U`	undoes the effect of the previous D, I, or R subcommand. Entering a U, carriage return, then another U cancels all previous subcommands for this REDO command and restores the line being corrected to its original form.

Once the command is corrected, enter a carriage return to issue the command.

Example

In the following example, the DISPLAY MAP command is corrected using the R subcommand.

```
>>DISPLAY MPA       ⇐ incorrect command entered

DISPLAY MPA
        ^
Syntax Error  (ATCERR 32435).
>>REDO              ⇐ request to REDO command
DISPLAY MPA         ⇐ command displayed for corrections
         RAP        ⇐ error replaced by correction
DISPLAY MAP         ⇐ corrected command displayed
```

SET SQLDBE

Defines the SQL DBEnvironment to be used by other IMAGE/SQL utility commands.

Syntax

SE[T] SQLDBE *DBEnvironment* [,MAINT=*DBE_MaintWord*]

Parameters

DBEnvironment	is the name of the DBEnvironment with which you want to work.
DBE_MaintWord	is the maintenance word for the DBEnvironment. This parameter can be omitted if you are an SQL database administrator (DBA).

Prerequisites

DBECreator status or DBEnvironment maintenance word specified as part of the command.

Description

Use the SET SQLDBE command to indicate the name of the SQL DBEnvironment with which you want to work. This name remains in effect until you exit the program or issue a new SET SQLDBE command. Note that because the IMAGE/SQL utility supports standard MPE/iX user, group, and account security rules, to issue this command for a DBEnvironment in a different group and account, be sure you have the correct capabilities.

To work with another DBEnvironment associated with the same TurboIMAGE/XL database, issue another SET SQLDBE command specifying the new DBEnvironment.

If the DBEnvironment does not exist, IMAGE/SQL displays this message:

```
DBE does not exist, do you want to create one? [Y/N] :
```

If you reply 'Y', a DBEnvironment and DBE files are automatically created for you. The files created are:

File Created	File Name
DBEConFile	*DBEnvironmentName*
DBEFile	*DBEnvironmentName*`FL`
LOG file	*DBEnvironmentName*`LG`
ATCINFO file	*DBEnvironmentName*`CR`

For example, if you issue the command

```
SET SQLDBE MYDBE
```

and the DBEnvironment MYDBE does not exist, these files are created: MYDBE, MYDBEFL, MYDBELG, and MYDBECR.

The size of MYDBEFL is 5000 pages, and that of MYDBELG is 10000 pages.

Example In the following example, the PARTSDBE DBEnvironment is specified and is in effect until you exit the IMAGE/SQL utility or issue another SET SQLDBE command.

```
>>SET SQLDBE PARTSDBE
>>
```

SET TURBODB

Defines the TurboIMAGE/XL database to be used by other IMAGE/SQL utility commands.

Syntax

```
SE[T] TURBODB TurboDB [,MAINT=TurboMaintWord]
```

Parameters

TurboDB	is the name of the TurboIMAGE/XL database with which you want to work.
TurboMaintWord	is the maintenance word for the TurboIMAGE/XL database. This parameter can be omitted if you are the TurboIMAGE/XL database creator (DBC).

Prerequisites

DBC status or TurboIMAGE/XL maintenance word specified as part of the command.

Description

Use the SET TURBODB command to indicate the name of the TurboIMAGE/XL database with which you want to work. This name remains in effect until you exit the program or issue a new SET TURBODB command. Note that because the IMAGE/SQL utility supports standard MPE/iX user, group, and account security rules, to issue this command for a TurboIMAGE/XL database in a different group and account, be sure you have the correct capabilities.

To work with another TurboIMAGE/XL database associated with the same DBEnvironment, issue another SET TURBODB command specifying the new database.

The SET BASE command is synonymous with the SET TURBODB. It has the same parameters and prerequisites.

Example

In this example, the SALES TurboIMAGE/XL database is specified.

```
>>SET TURBODB SALES
>>
```

SPLIT

Divides a mapped column into two or more smaller mapped columns.

Syntax

```
SP[LIT] MapTable.MapCol
INTO NewColSpec [,...]
```

Parameters

MapTable is the name of the mapped table containing the mapped column.

MapCol is the name of the mapped column to be split into smaller units.

NewColSpec is the specification of the new mapped column. Repeat this parameter for each new mapped column. The following syntax is used for the new mapped columns:

NewMapCol:SourceType[:*MappedType*]

NewMapCol is the new mapped column name. This name can be up to 20 bytes in length and can be made up of any combination of letters (A to Z), decimal digits (0 to 9), $, #, @, or _ (underscore). However, the first character cannot be a decimal digit or an underscore. (Note that lowercase letters are automatically converted to uppercase letters.)

SourceType is the TurboIMAGE/XL type the new mapped column would have if it were a data item in a TurboIMAGE/XL database.

MappedType is the new mapped column's SQL type. Refer to Table 2-6 for default and alternative data type mappings. If omitted, default type mapping is supplied.

Prerequisites

- SET SQLDBE issued.
- SET TURBODB issued.
- DBA authority.
- Database attached.

Description

Use the SPLIT command to divide mapped columns into smaller units. This is sometimes necessary because one data item can be used to contain information about several logically discrete units.

When a TurboIMAGE/XL database becomes part of a DBEnvironment, it is no longer necessary to keep logically separate information in one mapped column. The SPLIT command can be used to divide mapped columns of this kind into several smaller more logically discrete mapped columns.

Use the following guidelines when using the SPLIT command:

1. To use this command, you must first determine what each new mapped column's data type would be if it were a source field in a TurboIMAGE/XL database.
2. Each new mapped column must appear in the SPLIT command in the same order in which it is stored in the original source field.
3. Only certain data type conversions are allowed from TurboIMAGE/XL to SQL. Refer to Table 2-6 for recommended and alternative data type conversions.
4. The total length of the proposed new source fields must match the length of the original source field or an error message is issued.

It is desirable to split mapped columns before IMAGE/SQL users access the attached database because when a mapped column is split, all user-created views containing the mapped column are dropped.

When a mapped column is split, the definition for the mapped table to which this mapped column belongs is removed from the system catalog of the DBEnvironment. This also results in removal of definitions for hash, B-Tree, and third-party indices for the table. The new definition with additional columns for the mapped table is reentered in the system catalog of the DBEnvironment. In addition, new definitions for indices are also reentered in the system catalog of the DBEnvironment. (Definitions may not be the same as before.)

If you split a mapped column which represents a TurboIMAGE key item or search item, the definitions for hash or B-Tree (if they exist) indices on the mapped column are not entered in the system catalog. For an example, if a key item is split but the related search item of the detail data set is not, definitions for hash and implicit B-Tree indices on the mapped search item will be entered in the SQL catalog. However, definitions for unique hash index and B-Tree index on the mapped key item will not be entered.

Although the definitions for third-party indices are reentered in the system catalog, it is recommended that when you define the third-party index, you define the index on the whole item, and not substring especially starting from byte offset other than 1. For information on third-party indices, refer to your vendor's documentation.

Example

The mapped column INVENTORY.PART_INFO is of type CHAR(60) and contains the following units of information about parts in the order shown:

- A code identifying the part.
- The version number of the part.
- Brief listing of any special considerations regarding the part.

To split this mapped column into its logical units, issue the following command:

```
>>SPLIT INVENTORY.PART_INFO INTO PART_ID_CODE:X4,&
                                 PART_VERSION_NO:I1,&
                                 PART_NOTES:X54

Updated information in table INVENTORY.
>>
```

In the above example, INVENTORY.PART_INFO is split into three mapped columns. No alternative data type mapping exists for the specified data types. Therefore, it is not necessary to specify the :*MappedType* parameter because in this case the mapped types default to the following types:

- PART_ID_CODE of type CHAR(4)
- PART_VERSION_NO of type SMALLINT
- PART_NOTES of type CHAR(54)

The combined length of these three mapped columns equals the total length of the original mapped column, INVENTORY.PART_INFO.

UPDATE TYPE

Updates data type mapping information.

Syntax

$$\texttt{U[PDATE] TYPE}\left\{\begin{array}{l} \textit{SourceType}\ \texttt{IN}\left\{\begin{array}{l} * \\ \textit{MappedTable} \end{array}\right\} \\ \texttt{IN}\ \textit{MappedTable.Col} \end{array}\right\}$$

$$\texttt{[TO}\ \textit{NewMappedType}\texttt{]}$$

Parameters

SourceType	is a TurboIMAGE/XL data item type whose data type mapping information you want to update. Use either this parameter or the *MappedTable.Col* parameter.
∗ (asterisk)	indicates that you want to update data type mapping for a source data type in all mapped tables where it occurs. Use either this option or specify the individual table to be updated.
MappedTable	is the name of a mapped table containing a data type whose mapping you want to update. Use either this parameter or the asterisk (∗) option.
MappedTable.Col	is the name of a column in a specific mapped table whose data type mapping you want to update. Use either this parameter or the *SourceType* parameter.
NewMappedType	is the new data type you want to assign. If this parameter is omitted, default IMAGE/SQL utility data type mapping is used. (See Table 2-6 for allowed data type mappings.)

Prerequisites

- SET SQLDBE issued.
- SET TURBODB issued.
- DBA authority.
- Database attached.

Description

Use the UPDATE TYPE command to update the data type mapping information in one of the following situations:

- For all occurrences of a specified source data type in the database.
- For all occurrences of the source data type in a specified mapped table.
- For one occurrence of the source data type in a particular column of a specified table.

To return the TurboIMAGE/XL data type or mapped column to default IMAGE/SQL utility data type mapping values, omit the TO *NewMappedType* parameter.

Update data type mapping before IMAGE/SQL users access the database because when a mapped column is updated, all user-created views containing these mapped columns are dropped. When a mapped column type is updated, the definition for the mapped table to which this mapped column belongs is removed from the system catalog of the DBEnvironment. This also results in the removal of definitions for hash, B-Tree, and third-party indices for the table. The new definition for the mapped table is reentered in the system catalog of the DBEnvironment. In addition, new definitions for indices are also reentered in the system catalog of the DBEnvironment.

UPDATE TYPE stores the new data type mapping information in the ATCINFO file (*DBEnvironmentName*CR). The DISPLAY MAP command reflects the updated type information. Data type updates are restricted to those data types that have alternative data type mapping. The UPDATE TYPE command only affects the DBEnvironment named in the SET SQLDBE statement.

Table 2-2 in Chapter 2 summarizes IMAGE/SQL data type mapping defaults and alternatives.

Note

Updating a TurboIMAGE field X16 or K8 to an SQL DATE type means that data entered through SQL is not readable via pure TurboIMAGE/XL applications. You need to use an API routine as described in Appendix F, "Date/Time API."

Example

The first example shows how to convert fields in a data base to a decimal value with a varying number of decimal places. You can map type J2 to decimal type (10,n) where $0 \leq n < 10$.

- Use this command to remap all J2 fields in a data base to 0 decimal places:

```
>>UPDATE TYPE J2 IN * TO DECIMAL(10,0)
```

- Use this command to remap all J2 fields in a data set to 2 decimal places:

```
    >>UPDATE TYPE J2 IN tablename  TO DECIMAL(10,2)
```

- Use this command to remap one particular field in a data set to 1 decimal place:

```
    >>UPDATE TYPE IN table.column  TO DECIMAL(10,1)
```

In this second example, the UPDATE TYPE command is used to specify the alternative mapping ALLBASE/SQL CHAR(4). This is a byte-by-byte transfer (R2 is 4 bytes long). No data conversion is performed.

```
    >>UPDATE TYPE R2 IN CUSTOMER TO CHAR(4)
    Updated information in table CUSTOMER.
    >>
```

After the data type update, the DISPLAY MAP command reflects the change:

```
>>DISPLAY MAP CUSTOMER

TurboIMAGE/XL DB : SALES.SERED.ATC
DBEnvironment    : PARTSDBE.SERED.ATC
Owner Name       : SALES

MAPPED(SOURCE)   SOURCE            MAPPED       SOURCE     MAPPED
   TABLE         FIELD             COLUMN        TYPE       TYPE    NOTES
------------ ---------------- ------------------ ------- ----------- -----



CUSTOMER (CUSTOMER)
             CUSTOMER#        CUSTOMER#          J2      INTEGER
             LAST-NAME        LAST_NAME          X16     CHAR(16)
             FIRST-NAME       FIRST_NAME         X10     CHAR(10)
             INITIAL          INITIAL            U2      CHAR(2)
             STREET           STREET             X26     CHAR(26)
             CITY             CITY               X12     CHAR(12)
             STATE            STATE              X2      CHAR(2)
             ZIP              ZIP                X6      CHAR(6)
             CREDIT-RATING    CREDIT_RATING      R2      CHAR(4)     IU

NOTES:
  I: Imprecise(float)/Incompatible(others) mapping between source and
     mapped data types
  U: Source field has been updated
>>
```

In the above example, the NOTES column indicates that the data type mapping for CREDIT_RATING, the only mapped column whose source data type is R2, has been updated. The I indicates that the updated data type mapping is incompatible with the original data type.

UPDATE USER

Updates an IMAGE/SQL user's class number, password, and/or DBOPEN mode in a DBEnvironment.

Syntax

$$\texttt{U[PDATE] USER} \left\{ \begin{array}{l} \textit{User@Acct} \\ \textit{User.Acct} \end{array} \right\} \texttt{TO} \left\{ \begin{array}{l} \texttt{PASS=}\textit{Password} \\ \texttt{MODE=}\textit{ModeNumber} \\ \texttt{CLASS=}\textit{ClassNumber} \end{array} \right\}$$

```
|,...|
```

Parameters

User@Acct	is the name used to identify the user in the DBEnvironment. This name, referred to as the DBEUserID, is made up of the MPE/XL user and account name, connected with the period or @ symbol.
Password	is a password in the TurboIMAGE/XL schema. The new IMAGE/SQL user has the same access to the TurboIMAGE/XL data as the user class associated with this password. You must specify CLASS, PASS, or MODE parameter.
ModeNumber	is the DBOPEN mode with which the database is opened for this user. You must specify this parameter and/or the *ClassNumber* parameter. Valid DBOPEN modes are 1 through 8. If this parameter is omitted, the mode defaults to 5.
ClassNumber	is the user class number.

Prerequisites

- SET SQLDBE issued.
- SET TURBODB issued.
- DBC required.
- DBA authority.
- Database attached.

Description

Use the UPDATE USER command to update user information stored in the ATCINFO file. Information about the user password and the user's DBOPEN mode can be updated.

When a user password is updated, the associated DBEUserID (*User@Account*) is automatically deleted from the old group and added to a new group whose name is derived from the new user class given in the command. (For example, the password corresponding to user class 12 produces a group named *OwnerName_12*.) If this group does not exist, it is created.

To enforce TurboIMAGE/XL security, views are created for this group. These views limit this group's access to those data sets and data items defined for the corresponding user class/password in the TurboIMAGE/XL database schema.

Example

In the following example, user RYAN.ATC is updated to DBOPEN mode 1 with user class 13.

```
UPDATE USER  RYAN.ATC TO MODE=1, CLASS=13

ALLBASE/SQL group SALES_13 created.
View SALES.DATE_MASTER_V13 created.
View SALES.DATE_PRODUCT_V13 created.
View SALES.DATE_VENDOR_V13 created.
View SALES.DATE_INVENTORY_V13 created.
>>
```

XEQ

Executes IMAGE/SQL utility commands from a command file instead of the standard input device.

Syntax

```
X[EQ] FileName
```

Parameters

FileName	is the name of an ASCII file containing commands and parameters.

Prerequisites

None, but for successful execution of the command file, you must meet prerequisites of all commands in the command file.

Description

Use the XEQ command to execute IMAGE/SQL utility commands from a file instead of the standard input device. For example, if an ATCLOG file has been saved as a permanent file, it can be used to reissue the commands stored in the log file.

When an XEQ command is entered, the IMAGE/SQL utility reads the specified file and executes the commands until it reaches the end-of-file. When the end-of-file is reached, control returns to the original command input device (in session mode, the terminal, or in batch mode, the batch input device). Note that unless the command file contains an EXIT or QUIT command, you remain in the IMAGE/SQL utility when the XEQ command terminates.

During the execution of command files, what you see at the terminal depends on the setting of the ECHO command. When ECHO is on, the IMAGE/SQL utility displays lines at the terminal as they are read from the command files. Both comments and commands in the file are displayed. Regardless of the setting of the ECHO command, normal command output and error messages are displayed at the terminal.

In batch mode, if an error occurs during the execution of an XEQ file, the IMAGE/SQL utility closes the XEQ file and terminates the job.

Example

In the following example, the contents of the file DOATTACH are listed using the MPE/iX PRINT command. The XEQ command then executes the commands stored in the DOATTACH file. Because commands are not displayed as they are executed, it can be assumed that the ECHO command option is off.

```
>>:PRINT DOATTACH
SET TURBODB ORDER4        ⟸ contents of XEQ file
SET SQLDBE PARTSDBE       ⟸    "        "
ATTACH                    ⟸    "        "
QUIT                      ⟸    "        "
>>
>>XEQ DOATTACH     ⟸ XEQ command issued
Split 1 compound source field(s) (ATCWARN 32063).
Mapped 15 source table/source field name(s) (ATCWARN 32062).
Mapped 1 incompatible source type(s) (ATCWARN 32061).

END OF PROGRAM
:
```

5

IMAGE/SQL Locking

This chapter describes how IMAGE/SQL assigns locks on IMAGE/SQL tables and how IMAGE/SQL handles deadlocks.

Note

For detailed information on TurboIMAGE/XL and ALLBASE/SQL locking concepts, refer to the *TurboIMAGE/XL Database Management System Reference Manual* and the *ALLBASE/SQL Reference Manual*.

Lock Assignment

Locks are assigned to IMAGE/SQL tables in two ways. First, you can explicitly set a lock with the LOCK TABLE statement. Second, you can implicitly assign locks on PUBLIC tables depending on the operation, DBOPEN mode, and ALLBASE/SQL Isolation Level as shown in Table 5-1.

Table 5-1. Assigned Locks

IMAGE/SQL or ALLBASE/SQL Operation	TurboIMAGE/XL DBOPEN Mode	ALLBASE/SQL Isolation Level	Lock Assigned
Any operation that modifies the TurboIMAGE/XL table	1 through 4	Any isolation level	Exclusive data set lock for master dataset. Predicate level lock for detail dataset if the “where” clause is specified.
Read (SELECT)	1 through 4	RR, CS, or RC	Exclusive data set lock for master dataset. Predicate level lock for detail dataset if the “where” clause is specified.
		RU	None
	5 through 8	Any isolation level	None

The SQL statements that can modify TurboIMAGE/XL tables are INSERT, UPDATE, or DELETE.

The DBOPEN mode is assigned at ATTACH time. To see what mode is assigned, use the IMAGESQL DISPLAY USERS command. If you have DBA authority, you can change the mode with the UPDATE USER command.

The following are the isolation levels supported in ALLBASE/SQL and IMAGE/SQL.

RR	Repeatable Read. The transaction uses locking strategies to guarantee repeatable reads, the default isolation level.
CS	Cursor Stability. Transaction uses locking strategies to assure cursor-level stability only.
RC	Read Committed. Transaction uses locking strategies to retrieve only rows that have been committed by some transaction.
RU	Read Uncommitted. Transaction reads data without obtaining additional locks.

The isolation levels are established with a BEGIN WORK statement. Locks are released when a COMMIT WORK or ROLLBACK WORK statement is issued. Therefore, to release any locks, issue a COMMIT WORK statement as frequently as possible.

For more information on isolation levels, and how they do locking on ALLBASE tables, refer to the *ALLBASE/SQL Reference Manual*.

IMAGE/SQL Deadlocks

A deadlock occurs when two or more transactions are in a simultaneous wait state and each needs data that is held and locked by the other.

When a deadlock is encountered, the TurboIMAGE/XL intrinsic DBLOCK returns an error to IMAGE/SQL. IMAGE/SQL then aborts the IMAGE/SQL transaction that caused the deadlock. Transaction priorities are not considered in determining which transaction is aborted—the IMAGE/SQL transaction that caused the deadlock is aborted.

To avoid a deadlock situation during a connection, set the MAXIMUM TIMEOUT and DEFAULT TIMEOUT. You can set the DEFAULT and MAXIMUM TIMEOUT during the START DBE NEW or change the value using SQLUtil. If you do not set the timeout value, the default is unlimited time.

6

IMAGE/SQL Transactions

This chapter describes IMAGE/SQL transactions, repeatable reads, and IMAGE/SQL aborted transactions.

Note For detailed information on ALLBASE/SQL transactions, refer to the *ALLBASE/SQL Reference Manual*.

Definition

A transaction is a unit of work consisting of one or more SQL statements referencing one or more databases in a DBEnvironment. Work done within a transaction can be committed (made permanent) or undone (rolled back). A transaction is started with an implicit or explicit BEGIN WORK statement and is either committed with COMMIT WORK or rolled back with ROLLBACK WORK.

An IMAGE/SQL transaction is started by the first SQL statement that results in a modification or repeatable read of an IMAGE/SQL table.

Repeatable Read

A repeatable read guarantees that data pages (for ALLBASE/SQL) or an IMAGE/SQL dataset (for IMAGE/SQL) selected or updated by the current transaction are not changed by other transactions until the current transaction ends with a COMMIT WORK or ROLLBACK WORK statement.

For ALLBASE/SQL, a repeatable read is specified with the RR isolation level in the BEGIN WORK statement.

For IMAGE/SQL, a repeatable read occurs if both of the following conditions are met:

- The IMAGE/SQL database is opened in DBOPEN mode 1, 2, 3, or 4. The DBOPEN mode used by IMAGE/SQL is determined by the MODE attribute of the IMAGE/SQL user, which can be seen with the IMAGESQL DISPLAY USERS command, and modified with UPDATE USER.
- The transaction was started with a BEGIN WORK statement using the isolation level RR, or by selecting data or modifying the database without explicitly using the BEGIN WORK statement.

Restrictions

You can only modify one type of data in a transaction. That is, you can do the following within the same transaction:

- Read a TurboIMAGE/XL and/or ALLBASE/SQL database *and* modify a TurboIMAGE/XL database.
- Read a TurboIMAGE/XL and/or ALLBASE/SQL database *and* modify an ALLBASE/SQL database.

You cannot update both an ALLBASE/SQL database and a TurboIMAGE/XL database within the same transaction. Such an attempt will produce an error.

Within a transaction, avoid a terminal read with locks in effect (DBOPEN mode 1 through 4 and isolation level of RR, CS, or RC) because the database could be locked for a lengthy period of time. See Table 5-1 for details on locking assignment.

The limitation of transaction size depends on the Transaction Management (XM) limit and the DBE LOG SIZE limit. The XM limit for every transaction cannot exceed 4MBytes.

Note TurboIMAGE/XL activity is not logged to DBELOG.

IMAGE/SQL Aborted Transaction

If an error occurs during a modification within an IMAGE/SQL transaction, the transaction is aborted. While a transaction is in an aborted state, any locks acquired during the transaction are still held, and any work performed during the transaction remains in effect until a ROLLBACK WORK statement is executed. In an aborted transaction, all statements except ROLLBACK WORK are disallowed. SQL statements that result in TurboIMAGE/XL intrinsic calls will fail with TurboIMAGE/XL error -222.

A

IMAGE/SQL Error Messages

13501	MESSAGE	`Invalid TurboIMAGE database access. TurboIMAGE intrinsic` *intrinsic number*`. (DBERR 13501)`
	CAUSE	The TurboIMAGE/XL intrinsic named in the message has returned an error code (-21) indicating a bad password or maintenance word.
	ACTION	1. Make sure the user logon is a valid IMAGE/SQL user for this TurboIMAGE/XL database and the password for the user logon is correct. (Use the DISPLAY USERS command and DBUTIL's SHOW *dbname* PASSWORDS command.) 2. Verify the user's access to the data set(s) in the query by checking the read/write list for each data set in the TurboIMAGE/XL schema file. 3. If the password and read/write list security are correct, make sure the user is referencing the correct view. See Chapter 2, Task 3, for information regarding the views that the IMAGE/SQL utility creates and their correct usage.

13502	MESSAGE	`Command not allowed on TurboIMAGE table. (DBERR 13502)`
	CAUSE	This command is not supported by IMAGE/SQL.
	ACTION	Do not issue this command against a mapped table.

13503	MESSAGE	`TurboIMAGE database subsystem access has been disabled. (DBERR 13503)`
	CAUSE	Subsystem access of the TurboIMAGE/XL database has been disabled.
	ACTION	Contact your DBA to enable the subsystem access to the TurboIMAGE/XL database. Refer to the discussion of the DBUTIL SET command in the *TurboIMAGE/XL Database Management System Reference Manual*
13504	MESSAGE	`TurboIMAGE database has been disabled for access. (DBERR 13504)`
	CAUSE	Access to the TurboIMAGE/XL database has been disabled.
	ACTION	Contact your DBA to enable access to the TurboIMAGE/XL database. Refer to the discussion of the DBUTIL ENABLE command in the *TurboIMAGE/XL Database Management System Reference Manual*.

13505	MESSAGE	`IMAGE/SQL data conversion error: error` *ATC internal error number*`, set` *data set number*`, item` *data item number*`, column` *column number*`.` `(DBERR 13505)`
	CAUSE	Data conversion between TurboIMAGE/XL format and ALLBASE/SQL format caused the error. Either the source or the target length is not correct, or the internal conversion code stored in ATCINFO file is corrupted.
	ACTION	Identify the TurboIMAGE/XL data set and data item which cause the problem and use the IMAGE/SQL utility DISPLAY MAP command to find out the current mapping. Then use UPDATE TYPE command to map again. If the problem still exists, call the HP Response Center.

13506	MESSAGE	`IMAGE/SQL conversion overflow: set` *data set number*`, item` *data item number*`, column` *column number*`, record` *turboIMAGE record number*`. (DBERR 13506)`
	CAUSE	Data conversion from TurboIMAGE/XL format to ALLBASE/SQL format resulted in an overflow.
	ACTION	Identify the TurboIMAGE/XL record and the data item that caused the conversion overflow and use the IMAGE/SQL utility DISPLAY MAP command to find out the current mapping. Then, either use the IMAGE/SQL utility's UPDATE TYPE command to change the mapping or correct the TurboIMAGE/XL data for that record. The *data set number* is determined by the set's position in the SETS part of the TurboIMAGE/XL schema. The *data item number* is determined by the item's position in the ITEMS part of the TurboIMAGE/XL schema. The *column number* is determined by the column's position in the mapped table. The *record number* is the record number for the TurboIMAGE/XL data entry causing the conversion overflow.

13507	MESSAGE	`Invalid IMAGE/SQL data format: set` *data set number*`, item` *data item number*`, column` *column number*`, record` *turbo record number*`. (DBERR 13507)`
	CAUSE	Data conversion from TurboIMAGE/XL format to ALLBASE/SQL format for ZONED data type found an illegal overpunch in the data.
	ACTION	Identify the TurboIMAGE/XL record and the data items that contains the illegal overpunch, and use IMAGE/SQL utility's DISPLAY MAP command to find out the current mapping. Then, either use IMAGE/SQL utility's UPDATE TYPE command to change the mapping or correct the TurboIMAGE/XL data for that record. The *data set number* is determined by the set's position in the SET part of the TurboIMAGE/XL schema. The *data item number* is determined by the item's position in the ITEMS part of the TurboIMAGE/XL schema. The *column number* is determined by the column's position in the mapped table. The *record number* is the record number for the TurboIMAGE/XL data entry containing the illegal overpunch.

13508	MESSAGE	`IMAGE/SQL conversion truncated; set` *data set number*`, item` *data item number*`, column` *column number*`, record` *turboIMAGE record number*`. (DBERR 13508)`
	CAUSE	Data conversion between TurboIMAGE/XL format and ALLBASE/SQL format resulted in error. The source length is longer then target length, so the remaining byte(s) will be truncated.
	ACTION	Identify the TurboIMAGE/XL record and the data item that caused the conversion truncation and use the IMAGE/SQL utility DISPLAY MAP command to find out the current mapping. Then, either use the IMAGE/SQL utility's UPDATE TYPE command to change the mapping or correct the TurboIMAGE/XL data for that record.

13509	MESSAGE	`TurboIMAGE language id does not match the DBE language id. (DBERR 13509)`
	CAUSE	The TurboIMAGE language id and the DBE language id are not the same.
	ACTION	Set the TurboIMAGE language id and the DBE language id to the same value.

13510	MESSAGE	`ATCINFO file needs recovery. (DBERR 13510)`
	CAUSE	The ATCINFO file may be inconsistent.
	ACTION	Use the IMAGE/SQL utility RECOVER command to recover the ATCINFO file. See Chapter 2, Task 16, for a discussion of the RECOVER process.

13511	MESSAGE	`Update Statistics failed. IMAGE/SQL error` *error number*`; DBINFO error` *error number*`. (DBERR 13511)`
	CAUSE	A call to the TurboIMAGE/XL procedure DBINFO on the data set failed. The most likely reason is that the user does not have any access to the data set.
	ACTION	Check the *TurboIMAGE/XL Database Management System Reference Manual* for the DBINFO error number and make sure the user has read access to the TurboIMAGE/XL data set. Refer to the discussion of DBINFO modes in Chapter 5 of the *TurboIMAGE/XL Reference Manual*.
13551	MESSAGE	`Unexpected IMAGE/SQL runtime error` *error number*`,` *error number*`,` *error number*`,` *error number*`. (DBERR 13551)`
	CAUSE	While running IMAGE/SQL, the IMAGE/SQL utility's RECOVER command is in effect.
	ACTION	Wait until the RECOVER command is finished, then run IMAGESQL.

13552	MESSAGE	`IMAGE/SQL error` *error number*`;` `TurboIMAGE error` *error number*`;` `TurboIMAGE intrinsic` *intrinsic number*`, Auxiliary error` *error number*`. (DBERR 13552)`
	CAUSE	The TurboIMAGE/XL intrinsic indicated by the intrinsic number failed.
	ACTION	Check the *TurboIMAGE/XL Database Management System Reference Manual* and correct the problem. For example:

```
ATC error 129; TurboIMAGE error
-1; TurboIMAGE intrinsic 401,
Auxiliary error 93. (DBERR 13552)
```

is the result of a DBOPEN failure (intrinsic number 401) because the HPFOPEN or FOPEN intrinsic failed with error 93, which means "security violation." The action in this case would be to check and correct the MPE/iX group or account level security setup.

The IMAGE/SQL error number is for Hewlett-Packard internal use only.

Error codes returned by each TurboIMAGE/XL intrinsic are documented in Chapter 5 of the *TurboIMAGE/XL Database Management System Reference Manual* as part of the description of that intrinsic. The intrinsics and their associated numbers are listed below:

401	DBOPEN
402	DBINFO
403	DBCLOSE
404	DBFIND
405	DBGET
406	DBUPDATE
407	DBPUT
408	DBDELETE
409	DBLOCK
410	DBUNLOCK
411	DBCONTROL
412	DBBEGIN
413	DBEND
414	DBMEMO
415	DBMAINT
418	DBEXPLAIN
419	DBERROR
420	DBXBEGIN
421	DBXEND
422	DBXUNDO

The auxiliary error number is the lower level subsystem error number, such as the MPE/iX file system error number.

13553	MESSAGE	`IMAGE/SQL internal error` *error number*, *error number*, *error number*, *error number*. `(DBERR 13553)`
	CAUSE	An IMAGE/SQL internal error has occurred. The cause may be the allocation of various internal control blocks problems. The error numbers give more detail about where the problem happened in the IMAGE/SQL code.
	ACTION	Copy down the error numbers and call the HP Response Center.

13554	MESSAGE	`IMAGE/SQL internal error` *error number*, *error number*, *error number*, *error number*. `(DBERR 13554)`
	CAUSE	At connection time, the procedure call to open the ATCINFO file has failed.
	ACTION	The third error number contains the file system info number, the fourth error number contains the subsys number. Use those to figure out the reason why open of the ATCINFO file failed.

13555	MESSAGE	`IMAGE/SQL internal error` *error number*, *error number*, *error number*, *error number*. `(DBERR 13555)`
	CAUSE	Unexpected escape occurred in IMAGE/SQL. The first error number contains the escapecode.
	ACTION	Copy down the error numbers and call the HP Response Center.

IMAGE/SQL Warning Messages

32051	MESSAGE	`Database already attached (ATCWARN 32051).`
	CAUSE	The database was already attached prior to executing the latest ATTACH.
	ACTION	No action is necessary. This is an advisory message only.

32052	MESSAGE	`Database not attached (ATCWARN 32052).`
	CAUSE	You attempted to execute a command that requires the database to be attached.
	ACTION	Attach the database and re-execute the command that gave this warning.

32059	MESSAGE	`Exceeded 3996 bytes. Columns omitted in` *table name* `(ATCWARN 32059).`
	CAUSE	The maximum limit of column width in an ALLBASE/SQL table is 3996 bytes. This limit was exceeded while creating mapped tables, so one or more columns had to be omitted in the definition of mapped table *table name*.
	ACTION	Restructure the database so that the total length of fields in a data set does not exceed 3996 bytes.

32060	MESSAGE	`Exceeded 255 columns. Columns omitted in` *mapped table name* `(ATCWARN 32060).`
	CAUSE	The maximum number of columns in an ALLBASE/SQL table is 255. This limit was exceeded while creating mapped tables, so one or more columns had to be omitted in the definition of the specified mapped table.
	ACTION	Restructure the database so that the total number of fields in the data set does not exceed 255, or group one or more fields into a compound field. Refer to “Data Entries” in Chapter 2 of the *TurboIMAGE/XL Database Management System Reference Manual*.

32061	MESSAGE	`Mapped` *number* `incompatible/imprecise source type(s) (ATCWARN 32061).`
	CAUSE	An exact mapping was not found for the specified number of source fields.
	ACTION	No remedial action required. Use DISPLAY MAP to review all the mappings.

32062	MESSAGE	`Mapped` *number* `source table/source field name(s) (ATCWARN 32062).`
	CAUSE	The specified number of source data set names had to be mapped because characters not allowed by ALLBASE/SQL were encountered.
	ACTION	No remedial action required. Use DISPLAY MAP to review all the mappings.

32063	MESSAGE	`Split` *number* `compound source field(s) (ATCWARN 32063).`
	CAUSE	The specified number of compound source fields have been split into constituent components.
	ACTION	No remedial action is required. Use the DISPLAY MAP command to review all the mappings.
32066	MESSAGE	`Duplicate user name (ATCWARN 32066).`
	CAUSE	The user name you specified already exists in the ATCINFO file.
	ACTION	No action is necessary. This is an advisory message only.
32067	MESSAGE	`No database attached to the DBEnvironment (ATCWARN 32067).`
	CAUSE	No TurboIMAGE/XL database is attached to the DBEnvironment.
	ACTION	Attach a TurboIMAGE/XL database.
32068	MESSAGE	`Warning: command containing a maintenance word has been logged (ATCWARN 32068).`
	CAUSE	A command containing a TurboIMAGE/XL or ALLBASE/SQL DBEnvironment maintenance word was logged in ATCLOG.
	ACTION	Secure the ATCLOG file if security is a concern.

32069	MESSAGE	`Warning: command containing a password has been logged (ATCWARN 32069).`
	CAUSE	A command containing a TurboIMAGE/XL database password was logged in ATCLOG.
	ACTION	Secure the ATCLOG file if security is a concern.

File System Error Messages

32203	MESSAGE	`Privileged File Error (ATCERR 32203, FSERR` *error number*`).`
	CAUSE	The specified file did not have the expected file code.
	ACTION	Specify the correct file. For example, in the SET TURBODB command, only a TurboIMAGE/XL database name is valid.
32204	MESSAGE	*filename* `- Non Existent File (ATCERR 32204, FSERR` *error number*`).`
	CAUSE	The specified file does not exist.
	ACTION	Specify the correct file name. For further reference, look up the file system error number in the *MPE/iX Intrinsics Reference Manual*.
32205	MESSAGE	*filename* `- Duplicate Permanent File (ATCERR 32205, FSERR` *error number*`).`
	CAUSE	The specified file already exists in the permanent file domain.
	ACTION	Specify an unique file name.

32206	MESSAGE	*filename* - Duplicate Temporary File (ATCERR 32206, FSERR *error number*).
	CAUSE	The specified file already exists in the temporary file domain.
	ACTION	Specify an unique file name.
32207	MESSAGE	File System Error on *filename* (ATCERR 32207, FSERR *error number*).
	CAUSE	A file system error related to the specified file has occurred.
	ACTION	Look up the file system error number in the *MPE/iX Intrinsics Reference Manual*.
32208	MESSAGE	*filename* - Security Violation (ATCERR 32208, FSERR *error number*).
	CAUSE	A MPE/iX security violation occurred while doing an operation on the specified file.
	ACTION	Check your group/account capabilities and security access to the specified file.
32209	MESSAGE	*filename* - FWRITE Error. Condition code is CCL (ATCERR 32209).
	CAUSE	An error occurred when writing to the specified file.
	ACTION	If the specified file is an ATCLOG file, file equate ATCLOG to a different file. Persistent occurrence of this error should be reported to the HP Response Center.

32210	MESSAGE	*filename* `- End-of-File encountered (ATCERR 32210).`
	CAUSE	The physical end-of-file has been found in the specified file. If the ATCLOG file has been edited, this error can occur when the IMAGE/SQL utility attempts to write log records because some editors automatically insert end-of-file markers.
	ACTION	If this happens with the ATCLOG file, file equate ATCLOG to a new file and turn the logging on. If this happens with another file, contact the HP Response Center.
32211	MESSAGE	*filename* `- FREAD Error. Condition code is CCL (ATCERR 32211).`
	CAUSE	An error occurred when reading from the specified file.
	ACTION	Examine the file to determine the cause of the error, then rerun the program.
32212	MESSAGE	*filename* `- FLOCK Error. Condition code is CCL (ATCERR 32212).`
	CAUSE	Failed to FLOCK the specified file.
	ACTION	Examine the file to determine whether the file has been locked by another process, then rerun the program.
32213	MESSAGE	*filename* `- FUNLOCK Error. Condition code is CCL (ATCERR 32213).`
	CAUSE	Failed to FUNLOCK the specified file.
	ACTION	Contact the HP Response Center.

32214	MESSAGE	*filename* `- File being used by IMAGE/SQL Utility (ATCERR 32214).`
	CAUSE	The specified file is being used by the utility.
	ACTION	This occurs if you try to XEQ a file that is already opened by the IMAGE/SQL utility for logging. File equate ATCLOG to a different file.
32215	MESSAGE	`Cannot close` *filename* `(ATCERR 32215, FSERR` *error number*`).`
	CAUSE	The specified file could not be closed.
	ACTION	Examine the file to determine the cause of the error, then rerun the program.
32216	MESSAGE	`Error in CATREAD. Returned status is` *number* `(ATCERR 32216).`
	CAUSE	Error in reading a message from the IMAGE/SQL message catalog (`ATCUT000.PUB.SYS`).
	ACTION	Examine the file to determine the cause of the error, then rerun the program.
32217	MESSAGE	*filename* `- File name identifier > 8 chars (ATCERR 32217).`
	CAUSE	The file specified is more than eight characters long.
	ACTION	Specify a file name less than or equal to eight characters.

32218	MESSAGE	*user name* `- User name identifier > 8 chars (ATCERR 32218).`
	CAUSE	The user name specified is more than eight characters long.
	ACTION	Specify a user name less than or equal to eight characters.
32219	MESSAGE	*filename* `- Group name identifier > 8 chars (ATCERR 32219).`
	CAUSE	The group name in the specified file name is more than eight characters long.
	ACTION	Specify a group name less than or equal to eight characters.
32220	MESSAGE	*filename* `- Account name identifier > 8 chars (ATCERR 32220).`
	CAUSE	The account name in the specified file name is more than eight characters long.
	ACTION	Specify an account name less than or equal to eight characters.
32221	MESSAGE	*filename* `- Cannot set the creator (ATCERR 32221).`
	CAUSE	Failed to set the creator for the specified file.
	ACTION	Setting the creator of the TC file has failed. Please check the capability of the user for the group/account where the TurboIMAGE database resides.

DBCORE Errors

32301	MESSAGE	`Unexpected DBCORE error (ATCERR 32301, DBCORE` *error number*`,` *error number*`,` *error number*`,` *error number*`).`
	CAUSE	A DBCORE subsystem error has occurred. May be caused by using the ATTACH or DETACH commands when other processes are accessing the DBEnvironment.
	ACTION	Obtain exclusive access to the DBEnvironment before using the ATTACH or DETACH commands. Use the ISQL SELECT command to check if other users are accessing the DBEnvironment: `isql=> SELECT * FROM SYSTEM.USER;` If you were not using the ATTACH or DETACH commands, copy down the error numbers and contact the HP Response Center.

32306	MESSAGE	`Cannot read DBECon file (ATCERR 32306, DBCORE` *error number*`).`
	CAUSE	An error occurred when reading the DBECon file of DBEnvironment.
	ACTION	Contact the HP Response Center.

32307	MESSAGE	`Invalid DBECon file (ATCERR 32307, DBCORE` *error number*`).`
	CAUSE	The file specified is not a DBECon file.
	ACTION	Check the spelling specified in the SET SQLDBE or DISPLAY TURBODBS command.

32351	MESSAGE	`ALLBASE/SQL CONNECT command failed (ATCERR 32351).`
	CAUSE	A preprocessed CONNECT failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.

32352	MESSAGE	`ALLBASE/SQL RELEASE command failed (ATCERR 32352).`
	CAUSE	A preprocessed RELEASE failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.

32353	MESSAGE	`ALLBASE/SQL CREATE TABLE command failed (ATCERR 32353).`
	CAUSE	A preprocessed CREATE TABLE failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.

32354	MESSAGE	`ALLBASE/SQL BEGIN WORK command failed (ATCERR 32354).`
	CAUSE	A preprocessed BEGIN WORK failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.

32355	MESSAGE	`ALLBASE/SQL DROP TABLE command failed (ATCERR 32355).`
	CAUSE	A preprocessed DROP TABLE failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.
32356	MESSAGE	`ALLBASE/SQL COMMIT WORK command failed (ATCERR 32356).`
	CAUSE	A preprocessed COMMIT WORK failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.
32357	MESSAGE	`ALLBASE/SQL CREATE VIEW command failed (ATCERR 32357).`
	CAUSE	A preprocessed CREATE VIEW failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.
32358	MESSAGE	`ALLBASE/SQL ROLLBACK WORK command failed (ATCERR 32358).`
	CAUSE	A preprocessed ROLLBACK WORK failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.

32359	MESSAGE	`ALLBASE/SQL ADD user TO GROUP command failed (ATCERR 32359).`
	CAUSE	A preprocessed ADD user failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.
32360	MESSAGE	`ALLBASE/SQL REMOVE user FROM GROUP command failed (ATCERR 32360).`
	CAUSE	A preprocessed REMOVE user failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.
32361	MESSAGE	`ALLBASE/SQL CREATE GROUP command failed (ATCERR 32361).`
	CAUSE	A preprocessed CREATE GROUP failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.
32362	MESSAGE	`ALLBASE/SQL GRANT SELECT ON table command failed (ATCERR 32362).`
	CAUSE	A preprocessed GRANT SELECT failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.

32363	MESSAGE	`ALLBASE/SQL GRANT CONNECT TO user command failed (ATCERR 32363).`
	CAUSE	A preprocessed GRANT CONNECT failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.

32364	MESSAGE	`ALLBASE/SQL REVOKE SELECT FROM table command failed (ATCERR 32364).`
	CAUSE	A preprocessed REVOKE SELECT failed.
	ACTION	This message is followed by an ALLBASE/SQL error message. Refer to the *ALLBASE/SQL Message Manual* for more information.

32366	MESSAGE	`ALLBASE/SQL SETOPT ATC failed. (ATCERR 32366).`
	CAUSE	The statement is not supported with SETOPT.
	ACTION	Do not use the statement in SETOPT.

General Errors

32401	MESSAGE	`Unknown Error. Escapecode -` *number* `(ATCERR 32401).`
	CAUSE	An internal error has occurred.
	ACTION	Contact the HP Response Center.

32402	MESSAGE	`Bad Maintenance Word (ATCERR 32402).`
	CAUSE	An incorrect maintenance word was supplied.
	ACTION	Supply the correct maintenance word.
32403	MESSAGE	`Maintenance word not specified (ATCERR 32403).`
	CAUSE	A maintenance word is required, but was not specified.
	ACTION	Specify the maintenance word.
32404	MESSAGE	`Non creator access.  Command disallowed (ATCERR 32404).`
	CAUSE	The command cannot be executed by anyone other than the TurboIMAGE/XL database creator.
	ACTION	Have the TurboIMAGE/XL database creator execute the command.
32405	MESSAGE	`Column/Table/TurboIMAGE type not found (ATCERR 32405).`
	CAUSE	The column name, table name, or TurboIMAGE/XL type supplied in SPLIT/UPDATE TYPE does not exist.
	ACTION	Specify the correct information.
32407	MESSAGE	`Invalid password (ATCERR 32407).`
	CAUSE	The password specified in the ADD/UPDATE USER command could not be associated with any class defined in the database.
	ACTION	Specify the correct password.

32408	MESSAGE	`Invalid DBOPEN mode (ATCERR 32408).`
	CAUSE	The DPOPEN mode specified in ADD/UPDATE USER command is not valid.
	ACTION	Specify a DBOPEN mode greater than or equal to five and less than or equal to eight.
32409	MESSAGE	`Invalid user name (ATCERR 32409).`
	CAUSE	The user name is not in the correct format.
	ACTION	Specify the correct user name format *username@accountname*.
32410	MESSAGE	`Cannot create ATCINFO file (ATCERR 32410, FSERR` *error number*`).`
	CAUSE	Failed to create the ATCINFO file. This happens at the first attach to a SQLDBE when the ATCINFO file needs to be created.
	ACTION	You should not have a pre-existing file with actual file designator the same as that of ATCINFO. For further reference, look up File System *error number* in *MPE/iX Intrinsics Reference Manual*.
32411	MESSAGE	`Cannot open ATCINFO file (ATCERR 32411, FSERR` *error number*`).`
	CAUSE	Failed to open the ATCINFO file due to the specified file system error.
	ACTION	Look up the file system error number in the *MPE/iX Intrinsics Reference Manual*.

32412	MESSAGE	`Cannot purge ATCINFO file (ATCERR 32412, FSERR` *error number*`).`
	CAUSE	Failed to purge the ATCINFO file due to the specified file system error.
	ACTION	Look up the file system error number in the *MPE/iX Intrinsics Reference Manual*.
32413	MESSAGE	`Cannot close ATCINFO file (ATCERR 32413, FSERR` *error number*`).`
	CAUSE	Failed to close the ATCINFO file due to the specified file system error.
	ACTION	Look up the file system error number in the *MPE/iX Intrinsics Reference Manual*.
32417	MESSAGE	`Out of directory space in ATCINFO file (ATCERR 32417).`
	CAUSE	Too many databases are attached to one SQL DBEnvironment.
	ACTION	Detach one or more databases, or consider attaching the database to a different SQL DBEnvironment that has fewer databases attached to it.
32418	MESSAGE	`Out of free space in ATCINFO file (ATCERR 32418).`
	CAUSE	Too many databases are attached to one SQL DBEnvironment.
	ACTION	Detach one or more databases.

32419	MESSAGE	`Region is busy (ATCERR 32419).`
	CAUSE	The current TurboIMAGE/XL database is in use by some other process.
	ACTION	Wait a while and retry the command.
32421	MESSAGE	`ATCINFO procedure error (ATCERR 32421, ATCSTAT` *error number*`, FSERR` *error number*`).`
	CAUSE	An internal error has occurred while accessing the ATCINFO file.
	ACTION	Try RECOVER command in IMAGESQL utility. If the error still occurs, call the HP Response Center.
32422	MESSAGE	`Error in getting ATCU pointer (ATCERR 32422, ATCSTAT` *error number*`).`
	CAUSE	An internal error has occurred. May be caused by using the ATTACH or DETACH commands when other processes are accessing the DBEnvironment.
	ACTION	Obtain exclusive access to the DBEnvironment before using the ATTACH or DETACH commands. Use the ISQL SELECT command to check if other users are accessing the DBEnvironment: `isql=> SELECT * FROM SYSTEM.USER;` If you were not using the ATTACH or DETACH commands, copy down the error numbers and contact the HP Response Center.

32423	MESSAGE	`Region disabled. Detach before proceeding (ATCERR 32423).`
	CAUSE	A previous command or a system crash caused the information in the ATCINFO file to become unstable.
	ACTION	Detach and then attach the database. Then, retry the command. Caution: the detaching of a disabled region unmaps all customized mapping. The detach is required to “clean up” the environment.
32424	MESSAGE	`Owner name > 17 chars (ATCERR 32424).`
	CAUSE	An owner name longer than 17 characters was specified.
	ACTION	Specify an owner name less than or equal to 17 characters.
32425	MESSAGE	`Invalid TurboIMAGE type (ATCERR 32425).`
	CAUSE	The TurboIMAGE/XL data type specified is not valid.
	ACTION	Specify a valid TurboIMAGE/XL data type. For further reference, see the *TurboIMAGE/XL Database Management System Reference Manual*.
32426	MESSAGE	`Invalid Mapping (ATCERR 32426).`
	CAUSE	The data type mapping specified in the UPDATE TYPE or SPLIT command is not correct.
	ACTION	Specify a valid mapping; for further reference see Chapter 2, Tasks 4 and 5.

32427	MESSAGE	Maximum number of columns (255) exceeded (ATCERR 32427).
	CAUSE	The maximum number of columns in an ALLBASE/SQL table is 255. This limit was exceeded while trying to split an item.
	ACTION	Reduce the number of split fields.
32428	MESSAGE	Maximum length of columns (3996 bytes) exceeded (ARCERR 32428).
	CAUSE	The maximum limit of column width in an ALLBASE/SQL table is 3996 bytes. This limit was exceeded while trying to do SPLIT/UPDATE TYPE.
	ACTION	Make sure that the total width of mapped columns does not exceed 3996 bytes.
32429	MESSAGE	Illegal Mapping - component length mismatch (ATCERR 32429).
	CAUSE	The TurboIMAGE/XL data type length of the source field does not match the total of split (target) fields in the SPLIT command.
	ACTION	Make sure that the total length of split fields is equal to that of the source field.
32430	MESSAGE	*TurboIMAGE/XL database name* - no subsystem access (ATCERR 32430).
	CAUSE	The subsystem access to *TurboIMAGE/XL database* has been disabled.
	ACTION	Contact your DBA to enable the subsystem access to the TurboIMAGE/XL database.

32431	MESSAGE	*name* - `Invalid ALLBASE/SQL name (ATCERR 32431).`
	CAUSE	The name specified contains illegal characters or has an illegal format.
	ACTION	Specify a legal ALLBASE/SQL name. For further reference see the *ALLBASE/SQL Reference Manual*.
32432	MESSAGE	*name* - `Duplicate name (ATCERR 32432).`
	CAUSE	The specified name has already been defined.
	ACTION	Specify an unique name.
32434	MESSAGE	`MPE status error (ATCERR 32434, MPESTAT` *error number*, *error number*`).`
	CAUSE	A call to MPE/iX internal routines (getsir, relsir and set_kpo_pointer) has failed.
	ACTION	Contact the HP Response Center.
32435	MESSAGE	`Syntax Error (ATCERR 32435).`
	CAUSE	The command entered is syntactically wrong.
	ACTION	Use the HELP command to see the correct syntax for the command.
32436	MESSAGE	`TurboDB not set (ATCERR 32436).`
	CAUSE	The SET TURBODB command has not been issued.
	ACTION	Issue a SET TURBODB *turbodbname* command.

32437	MESSAGE	`SQLDBE not set (ATCERR 32437).`
	CAUSE	The SET SQLDBE *dbenvironment* command has not been issued.
	ACTION	Issue a SET SQLDBE *dbenvironment* command.
32438	MESSAGE	`ALLBASE/SQL DBE language does not match the TurboIMAGE language (ATCERR 32438).`
	CAUSE	The native language of the DBEnvironment and the TurboIMAGE/XL database do not match.
	ACTION	Contact your DBA to change the native language of either one of the databases so they match.
32439	MESSAGE	`Cannot map TurboIMAGE type P1. Attach aborted (ATCERR 32439).`
	CAUSE	A P1 TurboIMAGE/XL data type cannot be mapped into any SQL data type.
	ACTION	Avoid the P1 data type in your TurboIMAGE/XL database.
32440	MESSAGE	`ATCINFO file needs recovery (ATCERR 32440).`
	CAUSE	The ATCINFO file may be inconsistent.
	ACTION	Use IMAGE/SQL utility RECOVER command to recover the ATCINFO file. See Chapter 2, Task 16, for a discussion of the RECOVER command.

32441	MESSAGE	`User name not found (ATCERR 32441).`
	CAUSE	The user name specified in the DELETE/UPDATE USER command does not exist.
	ACTION	Specify the correct user name.
32442	MESSAGE	*TurboIMAGE/XL database* `- Access is disabled (ATCERR 32442).`
	CAUSE	Access to the TurboIMAGE/XL database has been disabled.
	ACTION	Contact your DBA to enable access to the TurboIMAGE/XL database. Refer to the discussion of the DBUTIL ENABLE command in the *TurboIMAGE/XL Database Management System Reference Manual*.
32443	MESSAGE	`Cannot recover ATCINFO file. Error` *error number*`,` *error number* `(ATCERR 32443).`
	CAUSE	An internal error has occurred.
	ACTION	Copy down the error numbers and call the HP Response Center.
32444	MESSAGE	*table/column name* `- Table/Column not found (ATCERR 32444).`
	CAUSE	The table or column name specified was not found in the ATCINFO file.
	ACTION	Check the spelling of the name you entered.

B

SALES Database Schema

The SALES database is used in the examples throughout this guide. The schema for this database is shown below:

```
BEGIN DATA BASE SALES ;
PASSWORDS:
  11  CREDIT  ;
  12  BUYER   ;
  13  dockhand;
  14  CLERK   ;
  18  manager ;

ITEMS:
  CITY             , X12(12,13,14/11);
  CREDIT-RATING    , R2(/14);
  CUSTOMER#        , J2;
  DATE             , X6;
  DELIVERED-DATE   , X6(/14);
  FIRST-NAME       , X10(14/11);
  INITIAL          , U2(14/11);
  LAST-NAME        , X16(14/11);
  LAST-SHIP-DATE   , X6(12/);
  LOCATION-BIN     , Z2(/13);
  ON-HAND-QTY      , J2(14/12);
  OTHER-VENDORS    , 3X16;
  PART-INFO        , X60;
  PRICE            , J2(14/);
  PRODUCT#         , U8;
  PRODUCT-DESCRIPT , X20;
  PURCHASED-DATE   , X6(11/14);
  QUANTITY         , I1(/14);
  STATE            , X2(12,13,14/11);
  STREET           , X26(12,13,14/11);
  TAX              , J2(14/);
  TOTAL            , J2(11,14/);
  UNIT-COST        , P8(/12);
  VENDOR           , X16(12,13/);
  ZIP              , X6(12,13,14/11);

SETS:

  NAME:    DATE-MASTER     ,A;
  ENTRY:   DATE             (3)                                    ;
  CAPACITY: 365(19);
```

```
   NAME:    CUSTOMER        ,M(14/11,18);
   ENTRY:   CUSTOMER#        (1)                                  ,
            LAST-NAME                                             ,
            FIRST-NAME                                            ,
            INITIAL                                               ,
            STREET                                                ,
            CITY                                                  ,
            STATE                                                 ,
            ZIP                                                   ,
            CREDIT-RATING                                         ;
   CAPACITY: 201(7);

   NAME:    PRODUCT         ,M(13,14/12,18);
   ENTRY:   PRODUCT#         (2)                                  ,
            PRODUCT-DESCRIPT                                      ;
   CAPACITY: 300(16);


   NAME:    VENDOR          ,M(13/12,18);
   ENTRY:   VENDOR           (1)                                  ,
            STREET                                                ,
            CITY                                                  ,
            STATE                                                 ,
            ZIP                                                   ;
   CAPACITY: 201(12);


   NAME:    INVENTORY       ,D(12,14/13,18);
   ENTRY:   PRODUCT#         ( PRODUCT                           ),
            ON-HAND-QTY                                           ,
            VENDOR           (!VENDOR                            ),
            OTHER-VENDORS                                         ,
            UNIT-COST                                             ,
            LAST-SHIP-DATE   ( DATE-MASTER                       ),
            LOCATION-BIN                                          ,
            PART-INFO                                             ;
   CAPACITY: 452(4);


   NAME:    SALES           ,D(11/14,18);
   ENTRY:   CUSTOMER#        ( CUSTOMER       (PURCHASED-DATE  )),
            PRODUCT#         (!PRODUCT                           ),
            QUANTITY                                              ,
            PRICE                                                 ,
            TAX                                                   ,
            TOTAL                                                 ,
            PURCHASED-DATE   ( DATE-MASTER                       ),
            DELIVERED-DATE   ( DATE-MASTER                       );
   CAPACITY: 504(14);

 END.
```

C

IMAGE/SQL and Database Utilities

DBUTIL

DBUTIL contains several enhancements to support IMAGE/SQL.

Displaying Information

If the TurboIMAGE/XL database which is also enabled for third-party indexing (TPI), is attached to any DBEnvironments, the ALL parameter of the SHOW command displays this information, as the following example illustrates:

```
:DBUTIL
HP30391C.05.09 TurboIMAGE/XL:DBUTIL (C)COPYRIGHT HEWLETT-PACKARD COMPANY

>>SHOW SALES ALL
  For database SALES

  Maintenance word is not present.

  Access is enabled.
      .     .     .
      .     .     .
  No other users are accessing the database.

  Attached to these HP SQL DBEnvironments:
    PARTSDBE.SERED.ATC
  Third Party Indexes are registered in these HP SQL DBEnvironments:
    PARTSDBE.SERED.ATC
>>
```

Purging an Attached Database

If you want to purge a TurboIMAGE/XL database that is attached to a DBEnvironment, a new parameter, DETACH, of the PURGE command allows you to detach and purge the database.

Detaching an Attached Database

If you want to detach a database from all DBEnvironments to which it is attached, the DETACH command allows you to do this as shown in this example:

```
:DBUTIL
HP30391C.07.00 TurboIMAGE/XL: DBUTIL (C) COPYRIGHT HEWLETT-PACKARD COMPANY

>>DETACH db1

  Database has been detached from these HP SQL DBEnvironments:
      RTDBE.BTRTESTS.IMAGESQL
      TPDBE.BTRTESTS.IMAGESQL
```

DBUTIL Enhancements for TPI

DBUTIL is enhanced for TPIs. The `ENABLE`, `DISABLE`, and `SHOW` statements have been modified for TPI registration as follows:

- `ENABLE` *database* `FOR INDEXING`

  If your database is already attached to at least one DBE and you enable it for indexing (TPIs must be configured before you use the `ENABLE` command), an attempt is made to enter definitions for the TPIs in all DBEs to which the database is attached. If the registration fails in one DBE, it proceeds to register in subsequent existing DBEs. If such an attempt fails in one or more DBEs, upon completion of the last attempt, a message is displayed as follows:

  ```
  Registration of Third-Party indices failed
    from these HP SQL DBEnvironments
  ```

  The names of DBEs in which the TPI definitions failed follows the message.

- `DISABLE` *database* `FOR INDEXING`

  If definitions for TPIs are entered in the ALLBASE/SQL SYSTEM CATALOG, the DISABLE database for `INDEXING` command triggers removal of the TPI information from all DBEs to which the database is attached. The removal of TPI information from all DBEs is necessary, otherwise, results can be unpredictable at run-time. If removal of this information fails in one DBE, it proceeds with subsequent existing DBEs. If such an attempt fails in one or more DBEs, upon completion of the last attempt, a message is displayed as follows:

  ```
  Dropping of Third-Party Indices failed
    from these HP SQL DBEnvironments
  ```

  This is followed by names of DBEs in which the dropping of TPIs failed.

  In this case, the database still remains `ATTACH`ed, however, the ALLBASE/SQL Optimizer has no knowledge of the TPIs as that information is removed.

  In brief, `DETACH` (of IMAGESQL) only deletes information about TPIs from one DBE. `DISABLE` (of DBUTIL) attempts to remove information on TPIs from all DBEs to which the database is attached as the database is turned off for TPI.

- `SHOW` *database* `FLAGS` or `SHOW` *database* `ALL`

  If definitions for the TPIs are registered in one or more DBEs, the `FLAGS` option displays a message indicating that the definitions of TPIs are entered. The `ALL` option displays all DBEs to which the database is attached, as well as the DBEs in which the TPI definitions are entered.

SQLUtil

SQLUtil contains two enhancements to support IMAGE/SQL:

- The ALL option of the SHOWDBE command displays the name of the ATCINFO file if any TurboIMAGE/XL databases are attached.
- If any TurboIMAGE/XL databases are attached and you purge the DBEnvironment, the PURGEDBE command automatically detaches them.

The following example illustrates these enhancements:

```
:SQLUTIL

                                           SAT, MAY 12, 1990,  2:59 PM
HP36216-02A.21.05             DBE Utility/3000             HP SQL/XL
(C)COPYRIGHT HEWLETT-PACKARD CO. 1982,1983,1984,1985,1986,1987,1988,
1989. ALL RIGHTS RESERVED.

>>SHOWDBE
DBEnvironment Name: PartsDBE
Maintenance Word: usr
Output File Name (opt): Carriage Return
-> ALL

Maintenance word:
DBEnvironment Language:  NATIVE-3000
AutoStart:  ON
User Mode:  MULTI
DBEFile0 Name:  DBEFILE0
Log File Name(s):  DBELOG1
Archive Mode:  OFF
DDL Enabled:  YES
No. of Runtime Control Block Pages:  37
No. of Data Buffer Pages:  100
No. of Log Buffer Pages:  24
Max. Transactions:  2
TurboIMAGE Information File:  ATCINFO

-> EXIT

>> PURGEDBE
DBEnvironment Name: SALESDBE
Purge DBEnvironment (y/n)? Y

DBEnvironment purged.
>>
```

SQLGEN

SQLGEN is a utility program for database administrators that generates the SQL commands necessary to re-create all or part of a DBEnvironment. The output from SQLGEN is a command file (sometimes called a schema) that can be used as input to ISQL in re-creating database objects.

SQLGEN also contains some enhancements for IMAGE/SQL. The following commands include mapped table definitions if any TurboIMAGE/XL databases are attached to the DBEnvironment.

- GENERATE TABLES
- GENERATE

D

SQL Exceptions

When SQL is used to access a TurboIMAGE/XL database, certain statements and options are unavailable or do not behave exactly as described in the *ALLBASE/SQL Reference Manual*. This appendix lists the known exceptions and also lists the SQL statements without exceptions.

SQL Statements with Exceptions

ALTER TABLE

- ALTER TABLE cannot be used to add a column to an IMAGE/SQL table. For example, the command

  ```
  isql=> alter table sales.customer add (newcol float);
  ```

  results in the following error:

  ```
  Operation not allowed on non-SQL table.  (DBERR 2454)
  ```

- ALTER TABLE cannot add a UNIQUE constraint to an IMAGE/SQL table. The command

  ```
  isql=> alter table sales.customer
         add constraint unique (last_name);
  ```

  results in this error:

  ```
  Command not allowed on a TurboIMAGE table.  (DBERR 13502)
  ```

- The ALTER TABLE DROP CONSTRAINT specification is also not supported.
- ALTER TABLE cannot add a CHECK constraint to an IMAGE/SQL table. The command

  ```
  isql=> alter table sales.customer
         add constraint check (last_name < 'SMITH')
         constraint checkname;
  ```

  is not allowed.

- ALTER TABLE cannot add a referential constraint to an IMAGE/SQL table.

BEGIN ARCHIVE

SQL archive logging does not capture modifications made to IMAGE/SQL databases.

TURBOSTORE Online Backup requires special procedures when TurboIMAGE/XL databases (and therefore IMAGE/SQL databases) are in use during the store operation.

BEGIN WORK

IMAGE/SQL transactions are managed by using the TurboIMAGE/XL intrinsics DBXBEGIN and DBXEND. The BEGIN WORK statement does not invoke DBXBEGIN until the first modification or repeatable read is requested for an IMAGE/SQL table.

The priority specified in a BEGIN WORK statement is ignored by IMAGE/SQL. IMAGE/SQL uses priority 100 for all lock requests involving IMAGE/SQL tables.

If isolation level RR (repeatable read), CS (cursor stability), or RC (read committed) is specified and the IMAGE/SQL user has a MODE attribute of 1, 2, 3, or 4, all read operations within the transaction will be repeatable reads.

If isolation level RU (read uncommitted) is specified, all read operations within the transaction will be read-uncommitted.

CHECKPOINT

Contents of log buffers are written to the log file or files, but contain no IMAGE/SQL work.

Data buffers containing changed pages are written to DBEFiles, but contain no IMAGE/SQL data.

COMMIT ARCHIVE

SQL archive logging does not capture modifications made to IMAGE/SQL databases.

COMMIT WORK

The KEEP CURSOR option of the OPEN statement cannot be used with a cursor that references an IMAGE/SQL table. Because KEEP CURSOR does not apply to IMAGE/SQL tables, COMMIT WORK:

- releases all IMAGE/SQL locks acquired during the current transaction
- closes all cursors that reference IMAGE/SQL tables
- does not implicitly start a new IMAGE/SQL transaction

COMMIT WORK ends an IMAGE/SQL transaction by calling DBXEND and then calls DBUNLOCK to release all locks acquired during the transaction.

CREATE INDEX

CREATE INDEX cannot be used to create an index on an IMAGE/SQL table. The command

```
isql => create index sales.newindex on
 sales.customer (last_name asc);
```

invokes this error:

```
Operation not allowed on non-sql table.  (DBERR 2454)
```

CREATE SCHEMA

CREATE SCHEMA cannot be used to define an IMAGE/SQL database, but can be used to define a view.

CREATE TABLE

CREATE TABLE is not supported.

DECLARE CURSOR

A cursor that references an IMAGE/SQL table (in the *QueryEXPRESSION* or *StatementName* of the FOR clause) cannot be opened with the KEEP CURSOR option of the OPEN statement.

DELETE

If SQL detects an error during a DELETE statement that references an IMAGE/SQL table, the current transaction is aborted regardless of the setting of the SET DML ATOMICITY and SET CONSTRAINTS statements.

The set of rows to be affected by the DELETE statement is determined before any rule fires, and this set remains fixed until completion of the rule. If the rule adds to the set, the additional rows will not be deleted. If the rule deletes from the set, a TurboIMAGE/XL intrinsic error will result and the current transaction will be aborted.

If an error occurs during the processing of any rule considered during execution of this statement, the current transaction will be aborted.

DELETE WHERE CURRENT

This has the same considerations as DELETE.

DROP DBEFILE

DROP DBEFILE cannot be used to drop DBEFiles containing IMAGE/SQL objects.

DROP TABLE

DROP TABLE cannot be used to drop an IMAGE/SQL table. An IMAGE/SQL table can be dropped only by detaching the database with the IMAGE/SQL DETACH statement.

DROP VIEW Cannot be used to drop a view created by IMAGESQL. A view created by IMAGE/SQL can be dropped only by detaching the database with the IMAGE/SQL DETACH command.

INSERT If SQL detects an error during an INSERT statement that references an IMAGE/SQL table, the current transaction is aborted regardless of the setting of the SET DML ATOMICITY and SET CONSTRAINTS statements.

If an error occurs during the processing of any rule considered during execution of this statement, the current transaction will be aborted.

IMAGE/SQL columns are defined as NOT NULL with default values. Default values are based on IMAGE/SQL item types. If columns are omitted from the column name list of an INSERT statement, the missing columns will be defined using the default values. The default values are listed in Table 2-5.

LOCK TABLE A LOCK TABLE statement always provides an EXCLUSIVE lock to a TurboIMAGE/XL data set.

OPEN A cursor referencing an IMAGE/SQL table (in the *QueryExpression* or *StatementName* of the DECLARE CURSOR statement) cannot be opened with the KEEP CURSOR option.

IMAGE/SQL columns are defined as NOT NULL. The INDICATOR option of the OPEN statement therefore cannot be used with IMAGE/SQL.

REVOKE REVOKE cannot be used to revoke authorities granted by the IMAGE/SQL ATTACH statement.

ROLLBACK WORK Savepoints cannot be defined in an IMAGE/SQL transaction. The TO clause of ROLLBACK WORK therefore cannot be used with an IMAGE/SQL transaction.

SAVEPOINTS Savepoints cannot be defined in an IMAGE/SQL transaction.

SELECT IMAGE/SQL columns are defined as NOT NULL. The INDICATOR option of the SELECT statement therefore cannot be used with IMAGE/SQL.

SET CONSTRAINTS

If a modification error occurs within an IMAGE/SQL transaction, the current transaction is aborted regardless of the setting of SET CONSTRAINTS.

SET DML ATOMICITY

If a modification error occurs within an IMAGE/SQL transaction, the current transaction is aborted regardless of the setting of SET DML AUTOMICITY.

START DBE NEW

The default DBEFile size of 150 pages used by START DBE NEW may not be sufficient to allow a database to be attached with IMAGESQL.

TRANSFER OWNERSHIP

IMAGE/SQL objects created by the IMAGESQL ATTACH statement cannot be transferred to another owner.

UPDATE

If ALLBASE/SQL detects an error during an UPDATE statement that references an IMAGE/SQL table, the current transaction is aborted regardless of the setting of the SET DML ATOMICITY and SET CONSTRAINTS statements.

The set of rows to be affected by the UPDATE statement is determined before any rule fires, and this set remains fixed until completion of the rule. If the rule adds to the set, the additional rows will not be updated. If the rule deletes from the set, a TurboIMAGE/XL intrinsic error will result and the current transaction will be aborted.

If an error occurs during the processing of any rule considered during execution of this statement, the current transaction will be aborted.

IMAGE/SQL columns are defined as NOT NULL. The NULL option of the SET clause therefore cannot be used with IMAGE/SQL columns.

If an IMAGE/SQL column specified in an UPDATE statement corresponds to a search or sort item in a TurboIMAGE/XL detail data set, the database's Critical Item Update flag must be set to ON. If an IMAGE/SQL column specified in an UPDATE statement corresponds to a search item in a TurboIMAGE/XL detail data set and the corresponding master data set is a manual master, the new column value must already exist as a chain head in the master data set.

UPDATE WHERE CURRENT

This has the same considerations as UPDATE.

SQL Statements without Exceptions

The following SQL statements behave as documented in the *ALLBASE/SQL Reference Manual*:

- ADD DBEFILE
- ADD TO GROUP
- ALTER DBEFILE
- ASSIGNMENT (=)
- BEGIN
- BEGIN DECLARE SECTION
- CLOSE
- CONNECT
- CREATE DBEFILE
- CREATE DBEFILESET
- CREATE GROUP
- CREATE PROCEDURE
- CREATE RULE
- CREATE TEMPSPACE
- CREATE VIEW
- DECLARE
- DESCRIBE
- DISABLE RULES
- DISCONNECT
- DROP DBEFILESET
- DROP GROUP
- DROP INDEX
- DROP MODULE
- DROP PROCEDURE
- DROP RULE
- DROP TEMPSPACE
- DROP VIEW
- ENABLE RULES
- END DECLARE SECTION
- EXECUTE
- EXECUTE IMMEDIATE
- EXECUTE PROCEDURE
- FETCH
- GENPLAN
- GOTO
- GRANT
- IF
- INCLUDE
- PREPARE
- PRINT

- RAISE ERROR
- REFETCH
- RELEASE
- REMOVE DBEFILE
- REMOVE FROM GROUP
- RESET
- RETURN
- SET CONNECTION
- SET MULTITRANSACTION
- SET PRINTRULES
- SET USER TIMEOUT
- SQLEXPLAIN
- START DBE
- START DBE NEWLOG
- STOP DBE
- TERMINATE USER
- UPDATE STATISTICS
- VALIDATE
- WHENEVER
- WHILE

E

SQL Views for Indices

This appendix is new with the G.2 version of this manual. It includes information on executing SQLINSTL and ODBCVIEW, as well as information on these four views that are related to indices:

- SYSTEM.IMAGEKEY
- CATALOG.IMAGEKEY
- SYSTEM.TPINDEX
- CATALOG.TPINDEX

Information related to these views for B-Tree indices is also included:

- SYSTEM.INDEX
- CATALOG.INDEX

Whenever you ATTACH a database to a DBEnvironment, the definitions for TurboIMAGE/XL and third-party indices (TPIs) are entered in the system catalog of the DBEnvironment. The definitions of TPIs are propagated to all DBEnvironments the database is attached to. You can observe the indices in the SQL views.

Executing SQLINSTL and ODBCVIEW

If you are on a version of ALLBASE/SQL older than the G.2 version, you must execute the SQLINSTL.PUB.SYS and ODBCVIEW.PUB.SYS files for each of the existing DBEnvironments.

Read the SQLINSTL.PUB.SYS file on your system for precautionary measures before executing it. See the *ALLBASE/SQL Database Administration Guide* or the *System Software Maintenance Manual* for more information on using SQLINSTL.

To install views required by the ODBC client-server interface, execute ODBCVIEW.PUB.SYS file for each existing DBEnvironment. For example:

```
:ISQL
isql=> connect to 'testdbe';
isql=> start odbcview.pub.sys;
                    ...
isql=> commit work;
isql=> connect to 'rtdbe';
isql=> start odbcview.pub.sys;
isql=> commit work;
isql=> exit;
```

You may see errors if you are executing this file for the DBEnvironment for the first time. These errors may be ignored. You can execute this command for all DBEnvironments in one session.

The B-Tree indices can be observed in the existing views, SYSTEM.INDEX by the DBA and CATALOG.INDEX by others. However, four new views have been added to the SQL system catalog to enable the DBA and other users to view TurboIMAGE/XL indices (key items and search items) as well as TPIs made aware to the SQL Optimizer. The Optimizer determines the most efficient path to the desired data, such as the type of scan to use and the proper order of operations. The DBA should periodically issue an UPDATE STATISTICS command on each TurboIMAGE/XL data set to refresh the statistics which are used by the Optimizer to derive an access plan. After all statistics have been updated, the DBA should issue VALIDATE commands to revalidate any stored sections that may exist on the data sets.

```
isql=> update statistics for table music.albums;
isql=> update statistics for table music.composers;
isql=> validate all modules with autocommit;
isql=> validate all procedures with autocommit;
```

For more information, see the "Maintenance" chapter of the *ALLBASE/SQL Database Administration Guide*.

Views for TurboIMAGE/XL

There are four SQL views added for TurboIMAGE/XL per DBEnvironment for the G.1 version. These are the SQL views for TurboIMAGE/XL hash indices on key items and search items and for TPIs:

- SYSTEM.IMAGEKEY
- CATALOG.IMAGEKEY
- SYSTEM.TPINDEX
- CATALOG.TPINDEX

SYSTEM.IMAGEKEY and CATALOG.IMAGEKEY

The DBA can view all TurboIMAGE/XL hash indices associated with a database by examining SYSTEM.IMAGEKEY. Other users can view the TurboIMAGE/XL hash indices to which they have access by examining CATALOG.IMAGEKEY.

For example, the DBA can issue this command:

```
isql => select * from system.imagekey where owner='music';


 -----------------------------------------------------
 INDEXNAME        | TABLENAME    | OWNER | UNIQUE | ...
 -----------------------------------------------------
 ALBUMCODE_M1     | ALBUMS       | MUSIC |  1     | ...
 COMPOSERNAME_M1  | COMPOSERS    | MUSIC |  1     | ...
 SELECTIONNAME_A1 | SELECTIONS_A| MUSIC |  1     | ...
 ALBUMCODE_D1     | SELECTIONS   | MUSIC |  0     | ...
 SELECTIONNAME_D2 | SELECTIONS   | MUSIC |  0     | ...
 COMPOSERNAME_D3  | SELECTIONS   | MUSIC |  0     | ...
 ALBUMCODE_D1     | LOG          | MUSIC |  0     | ...
 SELECTIONNAME_D2 | LOG          | MUSIC |  0     | ...
```

Columns in SYSTEM.IMAGEKEY and CATALOG.IMAGEKEY

The following columns exist in both SYSTEM.IMAGEKEY and CATALOG.IMAGEKEY (PRIMARIES, SCCCOUNT, and DCCCOUNT are not calculated or used in G.2):

`INDEXNAME`	Name of the TurboIMAGE/XL item, plus a suffix. The following suffixes are used by IMAGESQL: _M1 is used when entering definition for hash index on manual master key item. _A1 is used when entering definition for hash index on automatic master key item. _D*n* is used when entering definitions for hash indices on detail data set search items, where *n* is the path number from 1 to 16.
`TABLENAME`	Name of the TurboIMAGE/XL data set on which the key is defined.

OWNER	Name of the TurboIMAGE/XL database (or the owner name used during the ATTACH) on which the key is defined.
UNIQUE	Uniqueness indicator: 0 if duplicates are allowed, that is, the index is not unique. 1 if duplicates are not allowed, that is, the key is unique. Hash index on the key item of a master data set (both automatic and manual) is always unique, except when defined on P and Z (decimal) data types. Hash indices on search items of detail data sets are always non-unique.
NUMC	Number of columns in the index. NUMC is always 1.
COLNUMS	A vector of 16 SYSTEM.COLUMN entries, which identifies the column numbers that make up the key. In ISQL, each column number is displayed as a field of 4 hexadecimal digits.
NDISTINCT	Number of distinct key values.
PRIMARIES	Number of primary slots used.
SCCCOUNT	Synonym Chain Cluster Count, which is a measure of how well the data is clustered on pages as the synonym chain (also known as the secondary chain) is traversed.
DCCCOUNT	Detail Chain Cluster Count, which is a measure of how well the data is clustered on pages as the detail chain is traversed.

SYSTEM.TPINDEX and CATALOG.TPINDEX

The DBA can view all TPIs associated with a database by examining SYSTEM.TPINDEX. Other users can view the TPIs to which they have access by examining CATALOG.TPINDEX. For example, the DBA can issue:

```
isql => select * from system.tpindex where owner='music';


 ------------------------------------------------
 INDEXNAME    | TABLENAME  | OWNER | UNIQUE | ...
 ------------------------------------------------
 ACODE_T1     | ALBUMS     | MUSIC |  0     | ...
 ALBUMC_T2    | SELECTIONS | MUSIC |  0     | ...
 SELECTNAME_T3| SELECTIONS | MUSIC |  0     | ...
 COMPOSER_T4  | COMPOSERS  | MUSIC |  0     | ...
```

Columns in SYSTEM.TPINDEX and CATALOG.TPINDEX

The following columns exist in both SYSTEM.TPINDEX and CATALOG.TPINDEX (NPAGES, NLEVELS, NLEAVES, NDISTINCT, NFIRST, NPERKEY, and CCOUNT are not calculated or used in G.2):

`INDEXNAME`	Name of the TPI. The following suffix is used by IMAGESQL when registering TPIs: _T*n* (Where *n* can be 1 to 400 depending on the TPIs that exist on the database.)
`TABLENAME`	Name of the data set on which the TPI is defined.
`OWNER`	Name of the TurboIMAGE/XL database (or the owner name used during the `ATTACH`) on which the TPI is defined.
`UNIQUE`	Uniqueness indicator: 0 if duplicates are allowed, that is, the index is not unique. 1 if duplicates are not allowed, that is, the index is unique.
`CLUSTER`	Clustering indicator: 0 if the index is not a clustering index. 1 if the index is a clustering index. TPIs are always registered as non-clustering.
`NUMC`	Number of columns in the index.
`COLNUMS`	A vector of 16 SYSTEM.COLUMN entries, which identifies the column numbers that make up the index. In ISQL, each column number is displayed as a field of 4 hexadecimal digits.
`NPAGES`	Number of pages containing the index.
`NLEVELS`	Number of levels in the TPI.
`NLEAVES`	Number of leaf pages in the TPI.
`NDISTINCT`	Number of distinct key values.
`NFIRST`	Number of distinct first key values.
`NPERKEY`	Number of pages per key.
`CCOUNT`	Cluster count, which indicates how well the data of the index are sorted: 0 before first UPDATE STATISTICS statement is processed. *n* (efficiency of clustering) best clustering if *n*=NPAGES of table indexed; worst if *n*=NROWS of table indexed.

CTIME	Time of creation: yyyymmddhhsstt.
COLDIRS	A vector of 16 direction entries, which indicates the direction of the corresponding column in the index definition. In ISQL, each column number is displayed as a field of 4 hexadecimal digits. 5 Ascending. 6 Descending.

SYSTEM.INDEX and CATALOG.INDEX

When you attach a database for which you have created one or more B-Tree indices on the key items of the master data sets, the ATTACH command enters definitions for the B-Tree indices on the key items and their related search items of the detail data sets. See the chapter, "B-Tree Indices," in the *TurboIMAGE/XL Database Management System Reference Manual* for more information on how to create B-Tree indices.

The definitions of B-Tree indices on key items, except P and Z types, of the master data sets are entered as UNIQUE indices. For the related search items of ALL of the related detail data sets, the definitions are entered as NON-UNIQUE indices.

View your new B-Tree indices in the existing views, SYSTEM.INDEX and CATALOG.INDEX. For example:

```
isql=> SELECT * FROM SYSTEM.INDEX WHERE OWNER = 'BTREE3';
Output will be truncated. (DBWARN 1)


select * from system.index where owner = 'BTREE3';
--------------------+--------------------+--------------------+------+-----
INDEXNAME           |TABLENAME           |OWNER               |UNIQUE|CLUST
--------------------+--------------------+--------------------+------+-----
TESTNAMEP16_B1      |PACKED              |BTREE3              |     0|
TESTNAMELENR2_B1    |REAL                |BTREE3              |     1|
TESTNAMELENR2_V1    |RDET1               |BTREE3              |     0|
TESTNAMELENR2_V1    |RDET2               |BTREE3              |     0|
TESTNAMELENR2_V1    |RDET3               |BTREE3              |     0|
TESTNAMEP16_V1      |PRZDET              |BTREE3              |     0|
TESTNAMELENR2_V2    |PRZDET              |BTREE3              |     0|
```

From the above example, you can see that there are two master data sets, PACKED and REAL, with B-Tree indices created on their key items. The remaining are related detail data sets.

Columns in SYSTEM.INDEX and CATALOG.INDEX

The following columns are in both SYSTEM.INDEX and CATALOG.INDEX:

`INDEXNAME`	Name of the TurboIMAGE/XL item, plus a suffix.

The following suffixes are used by IMAGE/SQL:

_B1	is used when entering the definition for B-Tree index on the master key item.
_V*n*	is used when entering the definition for B-Tree index on the related search item of detail data set, where *n* is the path number from 1 to 16.

All other column names are the same as for SYSTEM.TPINDEX and CATALOG.TPINDEX. Refer to "Columns in SYSTEM.TPINDEX and CATALOG.TPINDEX," earlier in this appendix.

F

Date/Time API

Date/Time API Description

The collection of conversion routines called the "Date/Time Application Programming Interface (API)" allows conversion of data to and from internal format of ALLBASE/SQL date/time values programmatically. You can enter dates, times, and intervals through the SQL interface in the format of the ALLBASE/SQL date/time data types and store it in the same format in your TurboIMAGE/XL database. This is particularly useful in a PC client/server environment where many of the PC tools have date/time data types which map directly to the ALLBASE/SQL date/time data types. However, you may want to read from and write to these fields using your TurboIMAGE/XL applications as well. In order to help you with that, the conversion routines of ALLBASE/SQL were made externally callable. They are the following:

- TO_DATE
- TO_TIME
- TO_DATETIME
- TO_INTERVAL
- TO_CHAR
- TO_INTEGER

The necessary information on this API is given later in this appendix. For additional information on ALLBASE/SQL date/time functions, refer to "Date/Time Functions" in the "Expressions" chapter of the *ALLBASE/SQL Reference Manual*.

Updating K8 Data Types

You can use K8 data type of TurboIMAGE/XL for ALLBASE/SQL date/time data types. The UPDATE TYPE command of IMAGE/SQL allows you to update your K8 data type of TurboIMAGE/XL to SQL data type DATE, TIME, DATETIME, or INTERVAL. The default K8 data type mapping to SQL is CHAR[16].

$$\texttt{U}\left[\texttt{PDATE}\right]\ \texttt{TYPE}\ \left\{\begin{array}{l} \textit{sourcetype}\ \texttt{IN}\ \left\{\begin{array}{l} * \\ \textit{mappedtbl}\end{array}\right\} \\ \texttt{IN}\ \textit{mappedtbl.mappedcol}\end{array}\right\}\left[\texttt{TO}\ \textit{newtype}\right]$$

where *newtype* can be DATE, TIME, DATETIME, or INTERVAL

Some examples are:

```
UPDATE TYPE IN table1.K8item1   TO DATE
UPDATE TYPE IN table2.K8item2   TO TIME
UPDATE TYPE IN table3.K8item3   TO DATETIME
UPDATE TYPE IN table4.K8item4   TO INTERVAL
UPDATE TYPE K8 IN *            TO DATE
```

After you update your K8 data type to one of the SQL date/time data types for SQL interface, you may want to read from and write to these K8 data type fields in your TurboIMAGE/XL database using TurboIMAGE/XL applications. The field is still K8 for TurboIMAGE/XL applications; it is updated to one of ALLBASE/SQL date/time types for the SQL interface.

Because the data is stored in the format of ALLBASE/SQL, it is necessary to convert it to a readable string after you retrieve the data. Similarly, it is necessary to convert character data into the ALLBASE/SQL date/time format before it can be inserted into your TurboIMAGE/XL database using the TurboIMAGE/XL intrinsics.

Use the new externally callable procedures to handle this conversion. This section describes the Date/Time API and how to use it.

Mapping of DATE/TIME Functions

There is a one-to-one mapping between the Date/Time API routines and the ALLBASE/SQL date/time functions as shown below:

Date/Time API routine	ALLBASE/SQL Date/Time function
DBTODATE	TO_DATE
DBTOTIME	TO_TIME
DBTODTTM	TO_DATETIME
DBTOITVL	TO_INTERVAL
DBTOCHAR	TO_CHAR
DBTOINT	TO_INTEGER

The functionality of the API routine is equivalent to its counterpart ALLBASE/SQL Date/Time function. The descriptions of the parameters for each of the Date/Time API routines follow.

DBTODATE

Syntax

```
DBTODATE (charval, stringlen, format, fmtlen, dateval, error)
```

Parameters

charval	4-byte address of an array of characters holding the character representation of a date.
stringlen	4-byte integer length of *charval* in bytes.
format	4-byte address of an array of characters holding the format specification of the input character string, *charval*. Only valid format specifications for the TO_DATE function are allowed.
fmtlen	4-byte integer length of *format* in bytes.
dateval	4-byte integer address of a 16-byte buffer in which the resulting date in ALLBASE/SQL internal format is stored.
error	4-byte address of a 4-byte integer where error code is returned. *error* is set to 0 if no error occurred. Otherwise, it is set to the DBERR code for the error returned.

DBTOTIME

Syntax

```
DBTOTIME (charval, stringlen, format, fmtlen, timeval, error)
```

Parameters

charval	4-byte address of an array of characters holding the character representation of time.
stringlen	4-byte integer length of *charval* in bytes.
format	4-byte address of an array of characters holding the format specification of the input character string, *charval*. Only valid format specifications for the TO_TIME function are allowed.
fmtlen	4-byte integer length of *format* in bytes.
timeval	4-byte integer address of a 16-byte buffer in which the resulting time in ALLBASE/SQL internal format is stored.
error	4-byte address of a 4-byte integer where error code is returned. *error* is set to 0 if no error occurred. Otherwise, it is set to the DBERR code for the error returned.

DBTODTTM

Syntax

```
DBTODTTM (charval, stringlen, format, fmtlen, dttmal, error)
```

Parameters

charval	4-byte address of an array of characters holding the character representation of date-time value.
stringlen	4-byte integer length of *charval* in bytes.
format	4-byte address of an array of characters holding the format specification of the input character string, *charval*. Only valid format specifications for the TO_DATETIME function are allowed.
fmtlen	4-byte integer length of *format* in bytes.
dttmal	4-byte integer address of a 16-byte buffer in which the resulting time in ALLBASE/SQL internal format is stored.
error	4-byte address of a 4-byte integer where error code is returned. *error* is set to 0 if no error occurred. Otherwise, it is set to the DBERR code for the error returned.

DBTOITVL

Syntax

```
DBTOITVL (charval, stringlen, format, fmtlen, itvlval, error)
```

Parameters

charval	4-byte address of an array of characters holding the character representation of interval value.
stringlen	4-byte integer length of *charval* in bytes.
format	4-byte address of an array of characters holding the format specification of the input character string, *charval*. Only valid format specifications for the TO_INTERVAL function are allowed.
fmtlen	4-byte integer length of *format* in bytes.
itvlval	4-byte integer address of a 16-byte buffer in which the resulting interval in ALLBASE/SQL internal format is stored.
error	4-byte address of a 4-byte integer where error code is returned. *error* is set to 0 if no error occurred. Otherwise, it is set to the DBERR code for the error returned.

DBTOCHAR

Syntax

`DBTOCHAR(`*dateval,datatype,format,fmtlen,charval,bufflen,error*)

Parameters

dateval	4-byte address of the 16-byte date, time, datetime, or interval value stored in the ALLBASE/SQL date/time format to be converted.
datatype	4-byte integer representing the data type of the input, DATEVAL. It must be one of the following values:

10	Date
11	Time
12	DateTime
13	Interval

format	4-byte address of an array of characters holding the format specification of the desired format for the character string result. Only valid format specifications for the TO_CHAR function are allowed.
fmtlen	4-byte integer length of *format* in bytes.
charval	4-byte address of a character buffer to put result. This routine will fill this buffer with the character string representation of the date/time value, blank-filling to the end of the buffer as indicated by the length. If the character string representation for the date/time value is longer than the specified length of the buffer, the character string will be truncated to specified length.
bufflen	4-byte integer length of *charval* buffer in bytes.
error	4-byte address of a 4-byte integer where error code is returned. *error* is set to 0 if no error occurred. Otherwise, it is set to the DBERR code for the error returned.

DBTOINT

Syntax

`DBTOINT(`*dateval, datatype, format, fmtlen, intval, error*`)`

Parameters

dateval	4-byte address of the 16-byte date, time, datetime, or interval value stored in the ALLBASE/SQL date/time format to be converted.
datatype	4-byte integer representing the data type of the input, DATEVAL. It must be one of the following values: 10 Date 11 Time 12 DateTime 13 Interval
format	4-byte address of an array of characters holding the format specification specifying which component (month, day, hour, etc.) of the input, *dateval*, should be converted to the integer. Only valid format specifications for the TO_INTEGER function are allowed.
fmtlen	4-byte integer length of *format* in bytes.
intval	4-byte address of a 4-byte buffer where the integer result gets stored.
error	4-byte address of a 4-byte integer where error code is returned. *error* is set to 0 if no error occurred. Otherwise, it is set to the DBERR code for the error returned.

Glossary

ATC

The acronym, ATC, stands for ALLBASE/Turbo CONNECT. The terms Turbo CONNECT and ALLBASE/Turbo CONNECT are synonymous with IMAGE/SQL. The acronym, ATC, appears in error messages for IMAGE/SQL (for example, ATCERR or ATCWARN).

ATCINFO

A permanent privileged file containing mapping information about data types and user security. By default, it is named *DBEnvironmentName*CR. If you want to set a file equation for this file, you must do so before attaching any TurboIMAGE/XL databases.

ATCLOG

A temporary unnumbered ASCII file. If IMAGE/SQL utility logging is on (the default), all IMAGE/SQL utility commands are written to this file. If it does not already exist, it is created. If it exists, log records are appended to it. By default, it is named ATCLOG. However, you can set a file equation for this file.

ATCUtil

Another name for the IMAGE/SQL utility program used in previous releases and in some error messages. The IMAGE/SQL utility program is now called **IMAGESQL**.

Attached Database

A TurboIMAGE/XL database whose data can be accessed through a DBEnvironment. Information about the attached TurboIMAGE/XL database is stored in the DBEnvironment.

B-Tree Index

A name used to refer to an index in IMAGE/SQL databases. It is the name for the technique of optimizing the search of an index by using a binary tree search. You can create an index only on the master key item. However, you are able to perform index searches using all of its corresponding detail data set search items as well. The master data set key item is an “explicit B-Tree index.” The corresponding detail data set search items are “implicit B-Tree indices.”

Column
The vertical component of a table.

DBA (Database Administrator)
A database administrator of the DBEnvironment. You must be a DBA of the DBEnvironment to which the TurboIMAGE/XL database is attached to issue most IMAGE/SQL utility commands. The creator of the DBEnvironment is automatically a DBA. Other ALLBASE/SQL users can be granted DBA authority by a DBA.

DBC (Database Creator)
The creator of the TurboIMAGE/XL database. You must be either the database creator or give the TurboIMAGE/XL database maintenance word to attach a database to a DBEnvironment. Commands that add users, or display or modify user information can only be executed by the DBC.

DBEConFile
The DBEnvironment configuration file. It contains startup parameters for the DBEnvironment. The contents of this file are initially determined at the time you issue the START DBE NEW command. The owner of this file is the DBECreator.

DBECreator
The individual who originally configured the DBEnvironment.

DBEnvironment
A collection of related files consisting of one or more ALLBASE/SQL databases that share the same logging and recovery process.

DBEnvironmentNameCR
The default name for the ATCINFO file is the first six characters of the *DBEnvironmentName* followed by CR. The ATCINFO file is a permanent privileged file containing mapping information about data types and user security.

DBEUserID
A name used to identify DBEnvironment users. It is made up of an MPE/iX user and account name connected with the @ sign.

DBNameTC
A permanent privileged file in the same group and account as the TurboIMAGE/XL database. It contains the fully qualified names of the DBEnvironments to which the TurboIMAGE/XL database is attached. This information is used to let utilities such as DBUTIL know that the database is attached to one or more DBEnvironments.

Detached Database
A TurboIMAGE/XL database whose data cannot be accessed through a DBEnvironment. No information about the TurboIMAGE/XL database is stored in the DBEnvironment. A TurboIMAGE/XL database must be detached from a DBEnvironment before it is restructured.

Explicit B-Tree Index
A B-Tree index created using DBUTIL or DBSCHEMA on the master data set key item. See "B-Tree Index."

IMAGE/SQL View
A view created by IMAGE/SQL based on mapped tables.

IMAGESQL
A utility program for IMAGE/SQL. It is another name for ATCUtil which was used in earlier releases and is still used in some current error messages.

Implicit B-Tree Index
A B-Tree index on the detail data set search item whose corresponding master key item has an explicit B-Tree index created. See "B-Tree Index."

ISQL
An interactive command processor that lets you enter SQL statements at the keyboard and observe query results, messages, and other information on a video display.

Jumbo Dataset
A data set greater than 4 GB.

Mapped Column
An ALLBASE/SQL column created in the ALLBASE/SQL DBEnvironment by the IMAGE/SQL utility from the source data set field. Characteristics of the source field are mapped by the IMAGE/SQL utility to ALLBASE/SQL characteristics.

Mapped Table
A table defined in the DBEnvironment based on a TurboIMAGE/XL data set. Data set characteristics are mapped by the IMAGE/SQL utility to ALLBASE/SQL characteristics. The naming convention for ALLBASE/SQL tables is *Owner.Table*. By default, IMAGE/SQL specifies the database name as the owner and the data set name as the table. Thus, the IMAGE/SQL convention for mapped table names is *MappedDatabaseName.MappedDataSetName*. At attach time you must substitute a different owner name if an already attached database has the same name.

Mapping
The process IMAGE/SQL uses to allow a TurboIMAGE/XL database to emulate a DBEnvironment database. Mapping takes place for TurboIMAGE/XL names, data sets, data items, data item types, and data security.

Native SQL Column
A column belonging to an ALLBASE/SQL table.

Native SQL Table
A table created by ALLBASE/SQL.

Owner
A term used to define ownership of ALLBASE/SQL tables, views, and other ALLBASE/SQL objects. For mapped tables, the owner is by default the TurboIMAGE/XL database name (*DatabaseName.Table*). If two or more TurboIMAGE/XL databases with the same name, but residing in different groups and accounts, are to be attached to the same ALLBASE/SQL DBEnvironment, an alternative owner name must be specified at ATTACH time.

Row
The horizontal component of a table.

Source Field
A data set field in an attached TurboIMAGE/XL database.

SQL Database
A logical entity consisting of all tables, views, and other SQL objects in a DBEnvironment having the same owner.

SQLCheck
A utility program to check the integrity of a DBEnvironment.

SQLGEN
A utility program for database administrators that generates the SQL commands necessary to re-create all or part of a DBEnvironment. The output from SQLGEN is a command file (sometimes called a schema) that can be used as input to ISQL in re-creating database objects.

SQLINSTL
A script to migrate between versions of minor releases such as from G1.14 to G1.15 or from G.0 to G.1.

SQLMigrate
A utility that lets you migrate a DBEnvironment between major releases of ALLBASE/SQL such as from F.0 to G.0. To use it, you must be the DBECreator or have SM capability. To run it, issue the command RUN SQLMIG.PUB.SYS.

SQLMON

A utility program to help you monitor DBEnvironment performance.

SQLUtil

A utility program that assists the database administrator with DBEnvironment maintenance, backup, and recovery. SQLUtil also lets you modify the startup parameters for a DBEnvironment. It is a different utility from the IMAGE/SQL utility.

SQLVer

A utility to check the version strings of the ALLBASE/SQL files.

System Catalog

An ALLBASE/SQL database of information about the DBEnvironment. It is owned by SYSTEM. It consists of several system views that contain data about the DBEnvironment. It differs from the DBECon file, which contains startup parameters, not definitions.

Table

The basic unit of storage in an ALLBASE/SQL database. Tables are made up of horizontal *rows* and vertical *columns* of data. The ALLBASE/SQL naming convention for a table is *Owner.TableName* where the owner is the creator of the table.

User-Created View

A view created by the IMAGE/SQL user on mapped tables or IMAGE/SQL views. This term is used to contrast these views with IMAGE/SQL views.

View

A table derived by placing a "window" over one or more tables. The derivation of a view is a SELECT command. View names are governed by the same rules as table names.

Index