Oracle® Migration Workbench

Reference Guide for Microsoft Access 2.0, 95, 97, 2000 Migrations

Release 9.2.0 for Microsoft Windows 98/2000 and Microsoft Windows NT

March 2002

Part Number: A97262-01

This reference guide describes how to migrate from a Microsoft Access database to an Oracle9*i* or Oracle8*i* database.

ORACLE

Oracle Migration Workbench Reference Guide for Microsoft Access 2.0, 95, 97, 2000 Migrations, Release 9.2.0 for Microsoft Windows 98/2000 and Microsoft Windows NT.

Part Number: A97262-01

Contents

5 Application Performance Tuning

A Microsoft Access Error Messages

B Code and Query Samples

Index

Send Us Your Comments

Oracle Migration Workbench Reference Guide for Microsoft Access 2.0, 95, 97, 2000 Migrations, Release 9.2.0 for Microsoft Windows 98/2000 and Microsoft Windows NT.

Part Number: A97262-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Email - infomwb_ww@oracle.com
- FAX - +353-1-803-1899
- Postal service:
  Documentation Manager
  Migration Technology Group
  Oracle Corporation
  Eastpoint Business Park
  Dublin 3
  Ireland

Preface

The Oracle Migration Workbench Reference Guide for Microsoft Access 2.0, 95, 97, 2000 Migrations provides detailed information about migrating a database from Microsoft Access 2.0, Microsoft Access 95, Microsoft Access 97, or Microsoft 2000 to Oracle9*i*, Oracle8*i*, Oracle8*i* Appliance, or Oracle8. This reference guide describes several differences between Microsoft Access and Oracle. It also outlines how those differences are dealt with by the Oracle Migration Workbench (Migration Workbench) during the conversion process.

This chapter contains the following sections:

Audience

This reference guide is intended for anyone who is involved in migrating a Microsoft Access database to Oracle using the Migration Workbench.

What You Should Already Know

You should be familiar with relational database concepts. You should also be familiar with the operating system environments where you are running Microsoft Access and Oracle.

How this Reference Guide is Organized

This reference guide is organized as follows:

Chapter 1, "Overview"

Introduces the Migration Workbench and outlines features of this tool.

Chapter 2, "Migration Process"

Outlines the architecture of Microsoft Access and Oracle, explains how to prepare the Microsoft Access database for migration, and describes how to migrate from Microsoft Access to Oracle using the Migration Workbench.

Chapter 3, "Data Types, Reserved Words, and Functions"

Illustrates the migration of data types and built-in functions from Microsoft Access to Oracle and provides a list of Oracle9*i* and Oracle8*i* reserved words.

Chapter 4, "Troubleshooting"

Provides tips on how to migrate hyperlinks and how to perform a manual migration if necessary

Chapter 5, "Application Performance Tuning"

Tips for increasing performance of the Microsoft Access application after you have migrated to Oracle.

Appendix A, "Microsoft Access Error Messages"

Provides a list of Microsoft Jet database engine errors.

Appendix B, "Code and Query Samples"

Uses code examples to illustrate the migration process.

How to Use this Reference Guide

Every reader of this reference guide should read Chapter 1, "Overview". This chapter provides an introduction to the concepts and terminology of the Migration Workbench.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at

http://www.oracle.com/accessibility/

Accessibility of Code Examples in Documentation

JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Related Documentation

For more information, see these Oracle Migration Workbench resources:

- Oracle Migration Workbench Frequently Asked Questions
- Oracle Migration Workbench Release Notes
- Oracle Migration Workbench Online Help

To download release notes, installation documentation, white papers, or other collateral, go to the Oracle Technology Network (OTN) Web site. You must register online before using OTN. You can register for free at:

http://otn.oracle.com/membership/index.htm

If you already have a user name and password for OTN, you can go directly to the Migration Workbench documentation section of the OTN Web site at:

http://otn.oracle.com/tech/migration/workbench

Conventions

This section describes the conventions used in the text and code examples of this documentation. It describes:

- Conventions in Text
- Conventions in Code Examples

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
Bold	Bold type indicates GUI options. It also indicates terms that are defined in the text or terms that appear in a glossary, or both.	The C datatypes, such as **ub4**, **sword**, or **OCINumber,** are valid. When you specify this clause, you create an **index-organized table**.
Italics	Italic typeface indicates book titles, emphasis, syntax clauses, or placeholders.	*Reference Guide* Run U*old_release*.SQL where *old_release* refers to the release you installed prior to upgrading.
UPPERCASE monospace (fixed-width font)	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, user names, and roles.	You can specify this clause only for a NUMBER column. You can back up the database using the BACKUP command.
lowercase monospace (fixed-width font)	Lowercase monospace typeface indicates executables and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, user names and roles, program units, and parameter values.	Enter sqlplus to open SQL*Plus. The department_id, department_name, and location_id columns are in the hr.departments table.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
Square Brackets [ ]	Indicates that the enclosed arguments are optional. Do not enter the brackets.	`DECIMAL (digits [ , precision ])`
Curly Braces { }	Indicates that one of the enclosed arguments is required. Do not enter the braces.	`{ENABLE | DISABLE}`
Vertical Line |	Separates alternative items that may be optional or required. Do not type the vertical bar.	`{ENABLE | DISABLE}` `[COMPRESS | NOCOMPRESS]`
`Ellipses ...`	Indicates that the preceding item can be repeated. You can enter an arbitrary number of similaritems. In code fragments, an ellipsis means that code not relevant to the discussion has been omitted. Do not type the ellipsis	`CREATE TABLE ... AS subquery;` `SELECT col1, col2, ... , coln FROM employees;`
Italics	Indicates variables that you must supply particular values.	`CONNECT SYSTEM/system_password`

`UPPERCASE`	Uppercase text indicates case-insensitive filenames or directory names, commands, command keywords, initializing parameters, data types, table names, or object names. Enter text exactly as spelled; it need not be in uppercase	`SELECT last_name, employee_id FROM employees;`
		`SELECT * FROM USER_TABLES;`
		`DROP TABLE hr.employees;`

`lowercase`	Lowercase words in example statements indicate words supplied only for the context of the example. For example, lowercase words may indicate the name of a table, column, or file.	`SELECT last_name, employee_id FROM employees;` `sqlplus hr/hr`

1

Overview

This chapter introduces the Oracle Migration Workbench (Migration Workbench). It contains the following sections:

Introduction

Products, such as Microsoft Access, allow developers and advanced users to build complete business systems. However, Microsoft Access is based on file sharing technology. Therefore, it lacks the speed, reliability, and robustness provided by an independent relational database management system (RDBMS) server.

Using an RDBMS server with Microsoft Access in a client/server architecture provides the strengths of both technologies. Microsoft Access provides excellent forms and reports systems, as well as a complete programming language (Microsoft Access Basic). An independent RDBMS provides reliable, robust, and secure high speed data management.

The Oracle RDBMS is a modern, scalable, high performance database server that can run on a wide range of computers from PCs to mainframes. Oracle operates in a networked, client/server environment. It can support thousands of simultaneous users, depending on the server.

This reference guide explains how to migrate from Microsoft Access 2.0, 95, 97, and 2000 to an Oracle9*i*, Oracle8*i*, Oracle8*i* Appliance, or Oracle8 database using the Migration Workbench. It also provides guidelines on how to modify Microsoft

Access applications to work with the new Oracle database. If you have an investment in Microsoft Access applications, you can retain this investment while adding the advanced features of Oracle to the application architecture.

Product Description

The Migration Workbench is a tool that simplifies the process of migrating the table data from a Microsoft Access database to an Oracle database. The Migration Workbench allows you to seemlessly migrate an entire application system, that is the database schema including validation rules, default values, indexes and relations, in an integrated, visual environment.

The Migration Workbench employs an intuitive and informative user interface (UI) and a series of wizards to simplify the migration process. To ensure portability, all components of the Migration Workbench are written in Java.

The Migration Workbench uses a repository to store migration information. This allows you to query the initial state of the application before migration. You can work independently of the production application. To work independently, you must load the components of the application system that the Migration Workbench can migrate into a Migration Workbench repository.

Features

The Migration Workbench release 9.2.0.1.0 is a wizard-driven tool. It is composed of core features and Microsoft Access migration specific features. The Migration Workbench allows you to:

- Migrate a complete Microsoft Access database to an Oracle database.
- Migrate tables (including validation rules and default values), indexes, relations, and primary keys.
- Continue to use the Microsoft Access forms and reports with the destination Oracle database through an ODBC connection.
- Migrate Microsoft Access databases with single-level linked Microsoft Access tables.
- Display a representation of the source database and its Oracle equivalent.
- Generate and view a summary report of the migration.
- Customize users, tables, indexes, and tablespaces.
- Customize the default data type mapping rules.

- Resolve conflicts, such as Oracle reserved words, automatically.
- Remove and rename objects in the Oracle Model.
- Migrate individual table data.
- Use the offline data loading capability to migrate large amounts of data to Oracle.

Glossary

The following terms are used to describe the Migration Workbench:

Application System is the database schema and application files that have been developed for a database environment other than Oracle, for example, Microsoft Access.

Capture Wizard is an intuitive wizard that takes a snapshot of the data dictionary of the source database, loads it into the Source Model, and creates the Oracle Model.

Dependency is used to define a relationship between two migration entities. For example, a database view is dependent upon the table it references.

Destination Database is the Oracle database to which the Migration Workbench migrates the data dictionary of the source database.

Migration Entity is an instance of a migration component. For example, the table `EMP` is a migration entity belonging to the table `MIGRATION COMPONENT`.

Migration Wizard is an intuitive wizard that helps you migrate the source database to Oracle.

Migration Workbench is the graphical tool that allows migration of an application system to an Oracle database environment.

Navigator Pane is the part of the Migration Workbench User Interface that contains the tree views representing the Source Model and the Oracle Model.

Oracle Model is a a series of Oracle tables that is created from the information in the Source Model. It is a visual representation of how the source database will look when generated in an Oracle environment.

Properties Pane is the part of the Migration Workbench User Interface that displays the properties of a migration entity that has been selected in one of the tree views in the Navigator Pane.

Progress Window is the part of the Migration Workbench User Interface that contains informational, error, or warning messages describing the progress of the migration process.

Software Development Kit (SDK) is a set of well-defined application programming interfaces (APIs) that provide services that a software developer can use.

Source Database is the database containing the data dictionary of the application system being migrated by the Migration Workbench. The source database is a database other than Oracle, for example, Microsoft Access.

Source Model is a replica of the data dictionary of the source database. It is stored in the Oracle Migration Workbench Repository and is loaded by the Migration Workbench with the contents of the data dictionary of the source database.

Workbench Repository is the area in an Oracle database used to store the persistent information necessary for the Migration Workbench to migrate an application system.

2

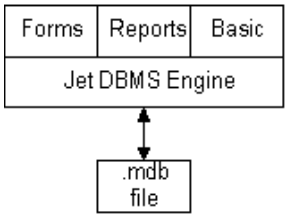
Migration Process

This chapter introduces the migration process by outlining the architecture of both Microsoft Access and Oracle. It includes the following sections:

Overview of the Microsoft Access Architecture

Microsoft Access is based on a file server DBMS technology named Microsoft Jet database engine. Forms, reports, and Basic code in Microsoft Access rely on Microsoft Jet database engine to manage data stored in the native .MDB file format. The following diagram illustrates the Microsoft Access architecture:

Figure 2–1 Microsoft Access Architecture



In a single-user Microsoft Access application, the .MDB file and Microsoft Access are located on the same system. In a multi-user Microsoft Access application, the .MDB file is placed on a file server and shared. Each client runs a copy of Microsoft Access and the Microsoft Jet database engine. In this configuration, Microsoft Jet database engine must move a large amount of data across the network, including whole tables, to complete its query processing.

The Data Access Objects (DAOs) DLL provides a hierarchy of classes to Microsoft Access Basic and Visual Basic. DAOs define and expose databases, workspaces, query definitions, parameters, recordsets, tables, fields, indexes, relationships, users, and groups from Microsoft Jet database engine. The Microsoft Access 2.0 DAO DLL is DA02016.DLL and the Microsoft Access 95 DAO DLL is DAO350.DLL.

Microsoft Jet database engine manages links to external data sources. Links to dBase, Paradox, and Btrieve are made through an internal Jet ISAM driver interface. These DLLs are included with Microsoft Access. Microsoft Jet database engine also supports a link to ODBC that allows access to a wide range of DBMS servers.

Microsoft Jet Database Engine Recordsets

When Microsoft Jet database engine processes a query, it returns a recordset (a cursor) for the resulting set of records. Microsoft Jet database engine supports snapshots and dynasets types of recordsets..

A snapshot is a picture of data as it existed at the time the query was run. When returning a snapshot, Microsoft Jet database engine runs the query to completion, extracts all of the resulting rows and columns into a virtual table, and presents this virtual table to the user. The user of a snapshot can perform a full range of operations on a snapshot, such as query the snapshot and base forms and reports on the snapshot. You cannot make changes to snapshot data. The snapshot data does not reflect changes made by other users after you have opened it.

A dynaset is a live view of the data. When returning a dynaset, Microsoft Jet database engine extracts the key values from the data and stores them in memory.

When a user requests rows of data from the dynaset, Microsoft Jet database engine fetches the rows of interest by looking them up in the base tables via the internally stored key values. Once you open a dynaset, the set of key values cannot change. Thus, while the data pointed to by the key value may change and is reflected to the user, new rows added after the query begins is not part of the set of key values and is not made visible to the user. Rows that you delete after you run the keyset query are still part of the set of key values. However, these rows are marked #DELETED# when displayed to the user.

The dynaset model is a powerful and flexible model that gives the user of PC-based information the opportunity to browse large quantities of data and update the data. When used with local data, dynasets are fast and effective. However, the dynaset model presents one of the key performance challenges when Microsoft Access works with an RDBMS server, such as Oracle.

Microsoft Jet Database Engine Multi-User Updates

Microsoft Jet database engine handles updates by multiple users by using optimistic and pessimistic locks.

Using pessimistic locking, Microsoft Jet database engine places a hard lock on the data page that contains the row being edited. Other users cannot start editing the locked row until the lock is abandoned or the changes are written to disk.

Microsoft Jet database engine employs an optimistic locking scheme when working with Oracle. An optimistic locking scheme does not place a hard lock on the source tables. Instead, when you want to commit a change, Microsoft Jet database engine checks to make sure that another user has not motified the data before it posts the changes.

Microsoft Jet Database Engine Enforced Referential Integrity

Microsoft Jet database engine supports declarative referential integrity. This includes primary key/foreign key relationships with one-to-one and one-to-many cardinality with cascading UPDATEs and DELETES.

Microsoft Jet Database Engine Query Processor

The Microsoft Jet database engine query processor does not support a full implementation of SQL. It optimizes queries, especially when the query references both local tables and remote tables. Microsoft Jet database engine can connect to a wide range of data sources and process queries against all of them. Transaction

support is limited to native file format database tables. Microsoft Jet database engine relies on the transaction support of any RDBMS attached via ODBC.

Microsoft Jet Database Engine Transactions

Microsoft Jet database engine supports an explicit transaction model. Transactions are not started until a `BeginTrans` statement is executed. Transactions are committed with `CommitTrans` and aborted with `Rollback`. In addition to using transactions to group units of work, you can use transactions to improve performance. If a program makes numerous references to a table, grouping the work in a single transaction forces Microsoft Jet database engine to perform the operation in memory. It then commits all work to disk when the transaction is committed. However, this type of transaction may not map directly to Oracle.

Overview of the Oracle Architecture

Oracle9*i*, Oracle8*i*, and Oracle8 databases are powerful, flexible, and scalable relational database management system (RDBMS) servers that run on a range of computer systems, from personal computers to largest mainframes. Oracle is designed to run effectively in a client/server environment and supports hundreds to thousands of users.

The Oracle architecture supports advanced server features, such as record locking with version references, (not page locking as provided by Microsoft Access), advanced query optimization, the PL/SQL programming language, data replication, distributed database management, and other important features.

The architectural features discussed here are only a few of the features found in Oracle. These features are focused on the elements that pertain to working with Microsoft Access. Refer to the Oracle Server guides for a complete description of the Oracle architecture. You can also view these guides in online format on CD-ROM. They are:

- *Oracle9i Database Concepts Release 1 (9.0.1)*
- *Oracle9i Database Administrator's Guide Release 1 (9.0.1)*
- *PL/SQL User's Guide and Reference Release 1 (9.0.1)*
- *Oracle9i Database Error Messages Release 1 (9.0.1)*

Triggers and Stored Procedures

Oracle allows you to write and store code in the DBMS along with data. You can associate trigger code with an UPDATE, INSERT, or DELETE event for each row or for an entire table. You can also set a trigger to run before an event or after an event. For example, you can set a trigger to run after a row is updated.

A stored procedure is a general routine, either function or subroutine, that is stored in pre-compiled form on the server. A trigger may call stored procedures, but triggers are only activated by specific database activity, such as the insertion of a row in a table.

When using Microsoft Access with Oracle, triggers and stored procedures play a role in mapping the functionality of Microsoft Access to Oracle, such as in the support for the Microsoft Access AUTONUMBER (COUNTER) data type in Oracle.

The Migration Workbench also converts certain Microsoft Access field validation rules to stored procedures within Oracle. For more information, see the Microsoft Access Functions topic.

PL/SQL Programming Language

The PL/SQL Programming Language is an ALGOL-based language, similar to Pascal. PL/SQL is a modern, full-featured programming language with exception handling. You can use PL/SQL to write stored programs and triggers in Oracle. It is also the programming language used in many of the client-side tools of Oracle, such as Forms from the Oracle Developer suite of products.

Sequences

A sequence is a unique number generator that is implemented in shared memory on a server. It is designed to provide a set of unique values for PL/SQL programs for use as primary keys. Sequences are designed for high performance applications that may otherwise single-thread on table-based unique number generators. You use sequences, along with supporting code in a trigger, to emulate the COUNTER field type in Microsoft Access.

Transactions

Unlike Microsoft Access, Oracle supports an implicit transaction model. Each SQL statement is part of a logical transaction. A logical transaction begins with the first SQL statement and ends with a Commit or Rollback statement. Immediately after

either of these statements, a new transaction takes effect with the next SQL statement.

Microsoft Access developers use transactions to improve the performance of Microsoft Jet database engine. Grouping database statements in a transaction forces Microsoft Jet database engine to attempt to complete all database work in memory. Microsoft Jet database engine defers writing to disk until the transaction is committed. When this use of transactions is mapped to Oracle through the Open Database Connectivity (ODBC), Microsoft Jet database engine sends only the outer most pair of `Begin` or `Commit` transaction requests. Oracle keeps an open transaction during the entire processing period. You must decide if you want this outcome when you move from Microsoft Access to Oracle.

Other Oracle Features

A database administrator has great flexibility when configuring Oracle. You can write data on multiple disks for increased performance. You can tune rollback and recovery options. You can allocate computer resources to optimize the configuration for each server. Oracle also supports distributed processing, so you can distribute data across multiple systems. Oracle offers a version of the server called Trusted Oracle Server for applications that require a higher level of user and use authentication.

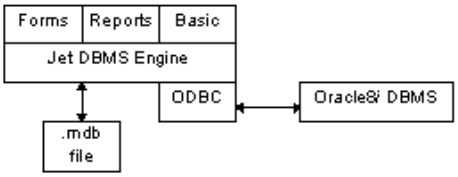
Comparing Microsoft Jet database engine, ODBC, and Oracle Architecture

Using Oracle with Microsoft Access can increase the robustness and reliability of a multi-user system. This reduces network traffic because only query requests and the resulting data are sent over the network, instead of entire tables. Microsoft Jet database engine technology is focused on single-user performance with adequate multi-user capabilities. Oracle is a mature central server technology focused on multi-user performance, rollback and recovery, and centralized query processing.

Obtaining adequate performance from the combination of Microsoft Access and Oracle requires an understanding of how Microsoft Jet database engine works with centralized servers.

The following diagram illustrates that Microsoft Access requires ODBC to make its connection to Oracle.

Figure 2–2 Microsoft Access/ODBC/Oracle Architecture



ODBC is an API that allows client applications to connect to different RDBMS servers. Microsoft Jet database engine has been designed to make efficient use of ODBC while requiring a level 1 ODBC driver. When Microsoft Access uses ODBC to connect to remote RDBMS servers, Microsoft Jet database engine continues to function as the DBMS engine for Microsoft Access. Microsoft Access forms, reports, and Basic code continue to work with Microsoft Jet database engine as if they were working with local or shared data in the .MDB file format. Microsoft Jet database engine presents remote Oracle tables as attached tables. These attached tables are created at design time and appear to be local tables.

Microsoft Jet database engine requires a primary key on tables in Oracle in order to support dynasets against those tables. If a remote table does not have a primary key, Microsoft Jet database engine only opens a snapshot on the table that cannot be updated.

It is possible to define tables in Microsoft Access that you can update, but do not have a primary key. When these tables are migrated to Oracle, they do not have a primary key and Microsoft Access is unable to update them. If you need to update the migrated tables, you should ensure that either all Microsoft Access tables that you can update have primary keys before you migrate or that you define a primary key once the tables are migrated to Oracle. Any tables that can be updated, but do not have primary keys are flagged with a warning in the Log window.

Preparing the Microsoft Access Database for Capture

It is recommended that you complete the steps below before you run the Capture Wizard. To prepare the Microsoft Access database for capture:

1. Make a back up of the Microsoft Access database files.
2. Turn off security. For more information, see the Turning Off Security Settings in Microsoft Access topic.

3. If the application contains linked tables to other Microsoft Access databases, refresh these links by opening the application in the Microsoft Access IDE and choosing **Tools -> Add Ins ->Linked Table Manager**.

> **Note:** The Migration Workbench supports linked tables to other Microsoft Access databases. The Migration Workbench automatically creates a user schema within the Oracle database for each Microsoft Access database migrated. However, Oracle recommends that you move all tables into a single Microsoft Access database in order to ensure that a single user schema is created in the Oracle database.

4. Compact the Microsoft Access database files by choosing **Tools -> Database -> Compact Database.**

> **Note:** Ensure that the database is not a replica database. The Migration Workbench cannot migrate a replica Microsoft Access database.

Turning Off Security Settings in Microsoft Access

The Migration Workbench does not support the migration of Microsoft Access databases that have security enabled. When you attempt to export the Microsoft Access database to XML using the Migration Workbench Exporter for Microsoft Access application, you may receive an error message if the database is secured. By default, the Migration Workbench uses the name of the Microsoft Access MDB file as the user name for the destination Oracle user. If you create an Oracle user in this way, the password is ORACLE.

In order to ensure that the Migration Workbench can migrate the Microsoft Access table data, it is necessary to copy the contents of the secured database into a new database. Everything is copied over to the new database, except for the security settings. You can then export the new Microsoft Access database to an XML file using the Oracle Migration Workbench Exporter for Microsoft Access.

To copy the contents of the secured database into a new database:

1. In Microsoft Access, choose **File** -> **New Database**.
2. Select the **Blank Database** icon and click **OK**.

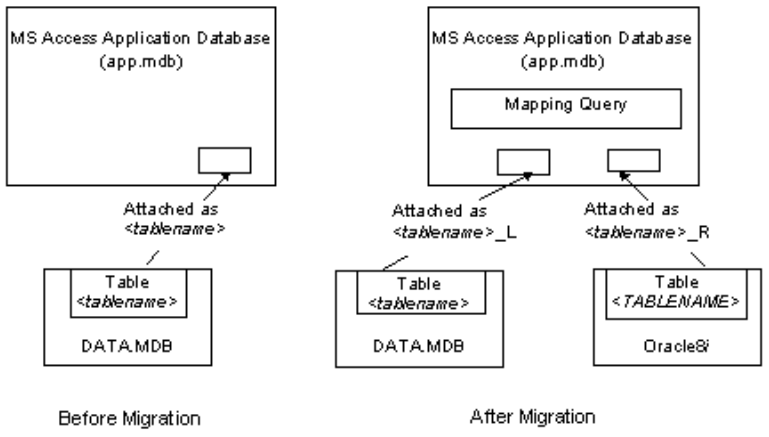
3. Specify the file name and location of the new database and click **Create**.
4. From within the new database, choose **File** -> **Get External Data** -> **Import**.
5. Select the secured Microsoft Access database you want to import and click **Import**.
6. From the Import Objects dialog, click **Options**. Ensure that the Relationships and Definition and Data options are selected.
7. From the Tables tab, choose **Select All**.
8. Click **OK**.
9. Export the new Microsoft Access database to an XML file using the Oracle Migration Workbench Exporter for Microsoft Access. If the database migrates successfully, the issue was due to a secured database.

Modifying the Microsoft Access Database

When you migrate an Microsoft Access database to Oracle, you must make some changes to allow the Microsoft Access forms and reports to continue to operate against the migrated data residing in the Oracle database. From the Migration Workbench, choose **Action** -> **Modify Access Database** to enable the Migration Workbench to make the necessary modification.

Figure 2–3 shows an Microsoft Access application before migration to Oracle and after migration to Oracle. Both before migration and after migration, app.mdb contains the forms, reports, macros, and Basic modules that make up the application. Before migration, app.mdb contains an attached table from data.mdb. This diagram uses <*tablename*> to refer to the name of the attached table.

Figure 2–3 Microsoft Access Application Before Migration and After Migration



After migration, app.mdb has two attached tables for each original table and query. The original Microsoft Access tables are renamed to <*tablename*>_L. The original table is exported to Oracle where it is called <*tablename*>. A table attachment is created to the Oracle table with the name <*tablename*>_R.

Because the forms, reports, and modules in app.mdb are expecting a table with the original table name, a query called <*tablename*> is created. This query takes the place of the original table in the application. The query can refer to either <*tablename*>_L or <*tablename*>_R. You can switch between the local and remote table as you move the application to Oracle. The query also helps resolve reserved word conflicts between Microsoft Access and Oracle. For example, you must rename the sequence column because it is an Oracle reserved word. The query can remap this Oracle column back to sequence for use by the Microsoft Access application. For a complete list, refer to the Oracle Reserved Words topic.

All requests for data from Microsoft Access forms and reports are directed to the query. This query references the generated ODBC linked table that retrieves the data from the Oracle database.

The original table that has been exported to Oracle is retained in case the data is needed locally. It is also retained so that any new Microsoft Access forms that you create can be based on this table. This enables the Microsoft Access forms to inherit

a complete set of table properties. After the Microsoft Access form is defined, you can switch the data source to the query so that Oracle can retrieve the data.

If the application opens a table directly (not using a dynaset or snapshot), it does not work with linked tables. This restriction also applies after you move the data to Oracle. If this happens, you may want to leave some tables in the app.mdb file so that each client has an independent copy. This could be appropriate for tables with lookup values, such as a State table. If you must move a table that is opened directly to the data.mdb file, you must change the application to use dynasets or snapshots.

ODBC Connection Issues with Linked Tables

When logging on to an Oracle Database through Microsoft Access the following Microsoft Jet database engine behavior should be considered.

When you connect to an Oracle server, the Microsoft Jet database engine caches the user name and password entered until Microsoft Access exits. Because of this connection caching, any attempt to connect to a linked table will fail, unless the user that corresponds to the cached user name has access privileges to that linked table. Connecting as a user that has system privileges or select privileges to all the tables ensures that all the tables are accessed from Microsoft Access.

Refer to the Managing User Privileges and Roles section in the Oracle9i Database Administrators guide for further details.

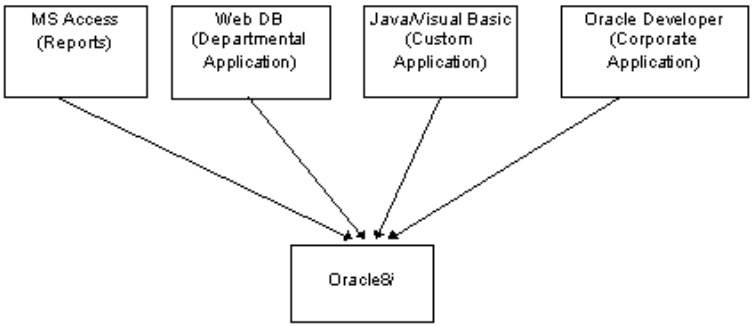
Extending the Application

After you move the data management portion of the Microsoft Access application to Oracle, you can rely on Oracle to protect the data and maintain all referential integrity and business rules that you have encoded in PL/SQL.

With this foundation, you can extend the application with Microsoft Access or a wide range of other tools. Oracle offers several high productivity tools such as Oracle Web DB, Oracle Developer, and Oracle Objects for OLE. Oracle Objects for OLE (OO4O) is a high performance connectivity solution for Visual Basic, Delphi, and other client tools that can control OLE Automation Servers.

In addition, if the application grows, you can move the Oracle server to larger computers without changing the application.

Figure 2–4 Extending the Application with a Mix of Client Tools



The *Oracle Objects for OLE and OLEDB/ADO Cookbook* provides additional information about how to extend the application. You can find this document on the Oracle Technology Network (OTN) Web site at:

http://otn.oracle.com/tech/migration/workbench

Using Oracle as the Back-end

You may notice some differences in behavior when using an Oracle database in conjunction with a Microsoft Access application. The differences are described in the following sections:

- Autonumber Options
- Default Options
- Multiple Forms Accessing the Same Record

Autonumber Options

When using a Microsoft Access table that has an AUTONUMBER column, the value of the AUTONUMBER is displayed once you start entering data for the record. Conversely, when you are using an ODBC link table referencing an Oracle table, the AUTONUMBER value displays after you have committed the record.

Default Options

When using a Microsoft Access table that contains defaults, the value of the default is displayed prior to entering data for the record. Conversely, when you are using an ODBC link table referencing an Oracle table, the default value displays after you have committed the record.

Multiple Forms Accessing the Same Record

When using local Microsoft Access tables, if two forms are accessing the same record, changes made in one form are immediately reflected in the other form. However, when you are using an ODBC link table referencing an Oracle table, modifications made in one form are reflected in the other form once you refresh the application by choosing the **Records** -> **Refresh** option.

3

Data Types, Reserved Words, and Functions

This chapter provides information about data types, reserved words, and functions. It includes the following sections:

Oracle Data Types

Table 3–1 describes the Oracle data types that the Migration Workbench supports:

Table 3–1 Supported Oracle Data Types

Data Type	Description
BLOB	A binary large object. Maximum size is 4 gigabytes.
CHAR	Fixed-length character data of length size bytes. Maximum size is 2000 bytes. Default and minimum size is 1 byte.
CLOB	A character large object containing single-byte characters. Both fixed-width and variable-width character sets are supported, both using the CHAR database character set. Maximum size is 4 gigabytes.

Table 3–1 Supported Oracle Data Types

Data Type	Description
DATE	The DATE data type stores date and time information. Although you can represent date and time information in both CHAR and NUMBER data types, the DATE data type has special associated properties. For each DATE value, Oracle stores the following information: century, year, month, day, hour, minute, and second.
FLOAT	Specifies a floating-point number with decimal precision 38, or binary precision 126.
LONG	Character data of variable length up to 2 gigabytes, or 231 -1 bytes.
LONG RAW	Raw binary data of variable length up to 2 gigabytes.
NCHAR	Fixed-length character data of length size characters or bytes, depending on the choice of national character set. Maximum size is determined by the number of bytes required to store each character, with an upper limit of 2000 bytes. Default and minimum size is 1 character or 1 byte, depending on the character set.
NCLOB	A character large object containing multibyte characters. Both fixed-width and variable-width character sets are supported, both using the NCHAR database character set. Maximum size is 4 gigabytes. Stores national character set data.
NUMBER	Number having precision p and scale s. The precision p can range from 1 to 38. The scale s can range from -84 to 127.
NVARCHAR2	Variable-length character string having maximum length size characters or bytes, depending on the choice of national character set. Maximum size is determined by the number of bytes required to store each character, with an upper limit of 4000 bytes. You must specify size for NVARCHAR2.
RAW	Raw binary data of length size bytes. Maximum size is 2000 bytes. You must specify size for a RAW value.
VARCHAR	The VARCHAR data type is currently synonymous with the VARCHAR2 data type. Oracle recommends that you use VARCHAR2 rather than VARCHAR.

For more information about Oracle data types, see the *Oracle9i SQL Reference, Release 1 (9.0.1)*.

Jet Data Types for Microsoft Access

Table 3–2 illustrates the Jet data types for Microsoft Access 97 and Microsoft Access 2000:

Table 3–2 Jet Data Types for Microsoft Access

Data Type	Description	Minimum	Maximum
Text	Stores variable length text	1	255
Memo	Large variable length text	1	64,000 bytes
Number Byte	1 byte storage	0	255
Number Integer	2 bytes storage	-32,768	32,767
Number Long Integer	4 bytes storage	-2,147,483,648	2,147,483,647
Number Single	4 bytes storage	-3.4×10^{38}	3.4×10^{38}
Number Double	8 bytes storage	-1.8×10^{308}	1.8×10^{308}
Currency	8 bytes storage - monetary values	-922337203685477.5808	922337203685477.5808
Counter Yes / No	4 bytes - AutoIncrement Field 1 bit storage - Boolean Value	0	2,147,483,647
Date / Time	8 bytes storage		
OLE Object	OLE, graphics other complex data	1	1.2 gigabytes

Default Data Type Mappings

Table 3–3 illustrates the default settings used by the Migration Workbench to convert data types from Microsoft Access to Oracle. The Migration Workbench allows you to change the default setting for certain data types by specifying an alternative type. You can do this in the Capture Wizard or in the Data Type Mappings page of the Options dialog box.

For more information about changing the default data type mappings, see the Oracle Migration Workbench Online Help.

Table 3–3 Default Data Type Mappings Used by the Migration Workbench

Microsoft Access Data Type	Oracle Data Type
Boolean	NUMBER(1, 0)
Byte	NUMBER(3, 0)
Currency	NUMBER(15, 4)
Date	DATE
Double	FLOAT(126)
Integer	NUMBER(5, 0)
Long	NUMBER(11, 0)
LongBinary	BLOB
Memo	CLOB
Single	FLOAT(126)
Text	VARCHAR2

Oracle Reserved Words

The following table lists words that are reserved in Oracle. The Migration Workbench appends an underscore to any object names that conflict with these reserved words.

ABORT	ACCEPT
ACCESS	ADD
ALL	ALTER
AND	ANY
ARRAY	ARRAYLEN
AS	ASC
ASSERT	ASSIGN
AT	AUDIT
AUTHORIZATION	AVG
BASE_TABLE	BEGIN
BETWEEN	BINARY_INTEGER
BODY	BOOLEAN
BY	CASE
CHAR	CHAR_BASE
CHECK	CLOSE
CLUSTER	CLUSTERS
COLAUTH	COLUMN
COMMENT	COMMIT
COMPRESS	CONNECT
CONSTANT	CRASH
CREATE	CURRENT
CURRVAL	CURSOR
DATA_BASE	DATABASE
DATE	DBA
DEBUGOFF	DEBUGON
DECIMAL	DECLARE

DEFAULT	DEFINITION
DELAY	DELETE
DESC	DIGITS
DISPOSE	DISTINCT
DO	DROP
ELSE	ELSIF
END	ENTRY
EXCEPTION	EXCEPTION_INIT
EXCLUSIVE	EXISTS
EXIT	FALSE
FETCH	FILE
FLOAT	FOR
FORM	FROM
FUNCTION	GENERIC
GOTO	GRANT
GROUP	HAVING
IDENTIFIED	IF
IMMEDIATE	IN
INCREMENT	INDEX
INDEXES	INDICATOR
INITIAL	INSERT
INTEGER	INTERFACE
INTERSECT	INTO
IS	LEVEL
LIKE	LIMITED
LOCK	LONG
LOOP	MAX
MAXEXTENTS	MIN
MINUS	MLSLABEL
MOD	MODE
MODIFY	NATURAL

NATURALN	NETWORK
NEW	NEXTVAL
NOAUDIT	NOCOMPRESS
NOT	NOWAIT
NULL	NUMBER
NUMBER_BASE	OF
OFFLINE	ON
ONLINE	OPEN
OPTION	OR
ORDER	OTHERS
OUT	PACKAGE
PARTITION	PCTFREE
PLS_INTEGER	POSITIVE
POSITIVEN	PRAGMA
PRIOR	PRIVATE
PRIVILEGES	PROCEDURE
PUBLIC	RAISE
RANGE	RAW
REAL	RECORD
REF	RELEASE
REMR	RENAME
RESOURCE	RETURN
REVERSE	REVOKE
ROLLBACK	ROW
ROWID	ROWLABEL
ROWNUM	ROWS
ROWTYPE	RUN
SAVEPOINT	SCHEMA
SELECT	SEPERATE
SESSION	SET
SHARE	SIGNTYPE

SIZE	SMALLINT
SPACE	SQL
SQLCODE	SQLERRM
START	STATEMENT
STDDEV	SUBTYPE
SUCCESSFUL	SUM
SYNONYM	SYSDATE
TABAUTH	TABLE
TABLES	TASK
TERMINATE	THEN
TO	TRIGGER
TRUE	TYPE
UID	UNION
UNIQUE	UPDATE
USE	USER
VALIDATE	VALUES
VARCHAR	VARCHAR2
VARIANCE	VIEW
VIEWS	WHEN
WHENEVER	WHERE
WHILE	WITH
WORK	WRITE
XOR	

Microsoft Access Functions

Default values and validation rules in Microsoft Access can contain Microsoft Access functions and operators. Corresponding functions and operators are generated as triggers in Oracle to support the operation of these default values and field validation rules whenever possible.

The Migration Workbench uses one of the following techniques to convert these functions and operators to Oracle:

Technique	Description
NO ACTION	There is a direct 1:1 mapping between the Microsoft Access function / operator and Oracle.
REPLACE IN PLACE	The Microsoft Access function / operator is directly replaced by the equivalent Oracle function / operator.
CODE	An Oracle function is written to duplicate the Microsoft Access functional capabilities. This function is created in an Oracle server during the migration.
*	Oracle does not handle this function. Therefore, you must change the Microsoft Access application so this function is not required in the default value or field validation rule.

Table 3–4 contains a list of Microsoft Access functions and explains how they are converted to Oracle.

Table 3–4 Conversion of Microsoft Access Functions to Oracle

Microsoft Access	Oracle	Conversion Action
Abs	Abs	NO ACTION
Asc	Ascii	REPLACE IN PLACE
Atn	-	*
CCur	CCUR	CODE
CDbl	CDBL	CODE
Chr	Chr	NO ACTION
Chr$	Chr	NO ACTION
CInt	CINT	CODE
CLng	CLNG	CODE

Table 3–4 Conversion of Microsoft Access Functions to Oracle

Microsoft Access	Oracle	Conversion Action
Command	-	*
Command$	-	*
Cos	COS	NO ACTION
CSng	To_Number	REPLACE IN PLACE
CStr	To_Char	REPLACE IN PLACE
CVar	To_Char	REPLACE IN PLACE
CVDate	-	*
Date	SYSDATE	REPLACE IN PLACE
Date$	SYSDATE	REPLACE IN PLACE
DateAdd	DATEADD	CODE
DateDiff	-	*
DatePart	-	*
DateSerial	-	*
DateValue	-	*
To_Date	-	*
Day	-	*
Environ	-	*
Environ$	-	*
Exp	EXP	NO ACTION
Fix	Trunc	REPLACE IN PLACE
Format	-	*
Format$	-	*
Hex	-	*
Hex$	-	*
Hour	-	*
In	-	CODE (not supported in def)
InStr	InStr	NO ACTION
Int	INTN	CODE
Is Not Null		CODE (not supported in def)
Is Null		CODE (not supported in def)
IsDate	-	*

Table 3–4 Conversion of Microsoft Access Functions to Oracle

Microsoft Access	Oracle	Conversion Action
LCase	LOWER	REPLACE IN PLACE
LCase$	LOWER	REPLACE IN PLACE
Left	LEFT	CODE
Left$	SUBSTR	CODE
Len	LENGTH	REPLACE IN PLACE
Like	*	
Log	LOG	NO ACTION
LTrim	LTRIM	NO ACTION
LTrim$	LTRIM	NO ACTION
Mid	SUBSTR	CODE
Mid$	MID	CODE
Minute	-	*
Month	-	*
Now	SYSDATE	REPLACE IN PLACE
Oct	-	*
Oct$	-	*
RGB	-	*
Right	RIGHT	CODE
Right$	SUBSTR	CODE
Rnd	RND	CODE
RTrim	RTRIM	NO ACTION
RTrim$	RTRIM	NO ACTION
Second	-	*
Sgn	SIGN	REPLACE IN PLACE
Sin	SIN	NO ACTION
Space	SPACE	CODE
Space$	LPAD / RPAD	CODE
Sqr	SQRT	REPLACE IN PLACE
Str	TO_Char	REPLACE IN PLACE
Str$	TO_Char	REPLACE IN PLACE
StrComp	-	*

Table 3–4 Conversion of Microsoft Access Functions to Oracle

Microsoft Access	Oracle	Conversion Action
String	LPAD / RPAD	CODE
String$	LPAD / RPAD	CODE
Tan	TAN	NO ACTION
Time	TIME_1	CODE
Time	TIME_2	CODE
Time$	SYSDATE	CODE
Timer	-	*
TimeSerial	-	*
TimeValue	-	*
Trim	TRIM	CODE
Trim$	TRIM	NO ACTION
UCase	UPPER	REPLACE IN PLACE
UCase$	UPPER	REPLACE IN PLACE
Val	TO_NUMBER	REPLACE IN PLACE
Weekday	-	*
Year	-	*

Table 3–5 contains a list of Microsoft Access operators and how they are converted to Oracle.

Table 3–5 Conversion of Microsoft Access Operators to Oracle

Microsoft Access	Oracle	Conversion Action
^	Power(m,n)	REPLACE IN PLACE
<	LTN	CODE
>	GTN	CODE
<=	LTEN	CODE
>=	GTEN	CODE
<>	NOTEQN	CODE
=	EQN	CODE
and	ANDN	CODE
or	ORN	CODE

Table 3–5 Conversion of Microsoft Access Operators to Oracle

Microsoft Access	**Oracle**	**Conversion Action**
not	NOTN	CODE
divide operator (/)	INTDIV	CODE
Eqv	-	*
Imp	-	*
Mod	MOD(m,n)	REPLACE IN PLACE
Xor	-	*

4

Troubleshooting

This chapter provides troubleshooting solutions. It instructs you through a manual conversion of a Microsoft Access database to Oracle. This guide assumes that you are starting with a separate application and data .MDB files, the app.mdb file and the data.mdb file. It instructs you on how to separate these files prior to performing a manual conversion.

This chapter includes the following sections:

Removing Microsoft Access Security

The Migration Workbench does not support the migration of Microsoft Access databases that have security enabled. In order to ensure that the Migration Workbench can migrate the Microsoft Access table data, it is necessary to copy the contents of the secured database into a new database. Everything is copied over to the new database, except for the security settings. You can then migrate the new database to Oracle.

To copy the contents of the secured database into a new database:

1. In Microsoft Access, choose **File** -> **New Database**.
2. Select the **Blank Database** icon and click **OK**.

3. From the File New Database dialog box, type a name for the database and click **Create**.
4. From within the new database, choose **File** -> **Get External Data** -> **Import**.
5. Select the secured Microsoft Access database you want to import and click **Import**.
6. From the Import Objects dialog, click **Options**. Ensure that the Relationships and Definition and Data options are selected.
7. From the Tables tab, choose **Select All**.
8. Click **OK**.

Erroneous Relations within Microsoft Access 95

There is a bug within Microsoft Access 95 that sometimes generates false relations. Since these erroneous relations do not appear in the Microsoft Access IDE user interface, you cannot delete them through the application because they are hidden from the user. The workaround is to create a new copy of the Microsoft Access database with no relations specified. You can then manually re-create the correct relations in the new database.

To copy the contents of the Microsoft Access database without specifying relations:

1. Create a new empty Microsoft Access database by choosing **File** -> **New Database**.
2. Select **Blank Database** and click **OK**.
3. From the File New Database dialog box, type a name for the database and click **Create**.
4. From within the new database, choose **File** -> **Get External Data** -> **Import**.
5. Select the original Microsoft Access database you want to import.
6. From the Import Objects dialog, select **Options**.
7. Deselect **Relationships** from the Import section, then click **OK**.

Note: If you want to migrate the legitimate relations from Microsoft Access, you must manually re-create them within the new database.

---

8. From the Tables tab, choose **Select All**.
9. Click **OK**.

Migrating Relations with Queries

The Migration Workbench currently does not support the migration of queries. Therefore, any relations consisting of queries do not migrate to Oracle. To avoid errors in migrating the Microsoft Access to Oracle, manually delete the relations by doing the following:

1. From within the Microsoft Access database, choose **Tools** -> **Relationships**.
2. Right-click on the line that shows the relationship, then select **Delete Relationship**.
3. Select **Yes** to confirm deletion.
4. Migrate the Microsoft Access database to Oracle.

Defining Primary Keys on a Table

If a table you specify does not have a defined primary key, you are unable to update the records of that table after migration. Microsoft Jet database engine requires a primary key on tables in Oracle in order to support dynasets against those tables. If the Oracle table does not have a primary key, Microsoft Jet database engine opens only a non-editable snapshot on the table. If you want to update the contents of the table through the Microsoft Jet database engine, you should do one of the following:

- Ensure that the original Microsoft Access table has a primary key defined on it. You must re-capture the Microsoft Access database to ensure the primary key is applied to the Oracle table.
- Define a primary key on the migrated Oracle table before you use the Migration Workbench to modify the Microsoft Access database. For example, you would use the following syntax to define a primary key:

  ```
  alter table <user>.<table> add ( constraint <primary key name> primary key (
  <column name> ));
  ```

  Therefore, if you substituted the variables user for Scott, table for Categories, primary key name for PK_CATEGORIES, and column name for ID, you would apply the following syntax:

```
ALTER TABLE Scott.Categories ADD ( CONSTRAINT PK_CATEGORIES PRIMARY KEY
( ID ) );
```

Migrating Table Data Using Microsoft Access ODBC Data Source

The following message displays during the data move of the migration process if the ODBC data source is not referencing a valid .mdb file:

```
The Microsoft Access ODBC Data Source Name "omwb_msaccess" is not configured
properly
```

When this message displays, perform the following procedures:

1. Choose Start->Settings->Control Panel.
2. Select ODBC Data Sources.
3. Select omwb_msaccess from the User DSN tab.
4. Click Configure.
5. Click Select in the Database section.
6. Browse to the database you want to migrate and click OK.

Incorrect Boolean Values When Migrating Data

When the Migration Workbench retrieves Boolean data, the Microsoft Access ODBC driver returns one of two values, 0 as FALSE or 1 as TRUE, therefore, FALSE and TRUE values are also represented in Oracle as 0 and 1.

However, an issue may arise when this data is used within a Microsoft VB application. Microsoft VB internally represents TRUE as -1 (instead of 1), and FALSE as 0. Therefore if you are evaluating based on TRUE, the logical programme behavior may be incorrect.

To ensure that the logic is unchanged, Oracle recommends that you update TRUE Boolean values in Oracle tables by executing the following command:

```
UPDATE <TABLENAME> SET <COLUMNNAME> = -1 WHERE <COLUMNAME> = 1
```

5

Application Performance Tuning

This chapter provides suggestions for tuning and customizing the way Oracle and Microsoft Access work together.

This chapter includes the following sections:

Selecting ODBC Drivers

There are a number of ODBC drivers available for Oracle. In addition to the driver supplied by Oracle, drivers are also available from Microsoft, Visigenic, MERANT, and others.

The performance of ODBC drivers can vary. If you are building a large-scale application, you need to profile the different ODBC drivers with the application. The best way to determine the performance is with the ODBC or OCI spy programs. These programs show you the calls that Microsoft Jet database engine makes to the ODBC API. They also show you the calls the Oracle ODBC driver makes against OCI.

Using Server-based Parameter Table

If you are administering an Oracle database that uses Microsoft Access as a front end, you can create a special parameter table in the Oracle database named

MSysConf to help you control communication between the Microsoft Access application and the Oracle database. When Microsoft Access first makes a connection to an Oracle database, it checks to see if the MSysConf table is present within the Oracle user schema that you are connecting to. You can use the MSysConf table to prevent storage of user logon information within a linked table, which increases the security of the application. You can also use the MSysConf table to optimize the record retrieval characteristics. The following table illustrates the structure of the MSysConf table within Oracle:

Column	Data Type
Config	NUMBER(10)
chValue	VARCHAR2(255)
nValue	NUMBER(10)
Comments	VARCHAR2(255)

The following table illustrates the Config and nValue column values that you can use to customize the way Microsoft Access interfaces with Oracle:

Config	nValue	Description
101	0	Do not allow the user to store the USERID and PASSWORD in linked tables.
101	1	Allow the user to store the USERID and PASSWORD in linked tables (default).
102	D	Delays D seconds between each retrieval.
103	N	Fetches N rows for each retrieval.

It is recommended that you create an MSysConf table in each Oracle database even if you plan on using the defaults. This way you can change the values in the table, rather than remember how to create and name the table at a later time.

Improving Application Startup Performance

You can tune Microsoft Access to speed up the process of establishing an ODBC connection at application startup time. When Microsoft Access opens a connection to an ODBC database, it determines the level of functionality provided by the particular ODBC driver. If you are relying on Oracle to provide full security, you

can bypass attempts by Microsoft Access to login to Oracle using the user, group, and password information of Microsoft Access.

The value of the `TryJetAuth` attribute determines whether login authentication is bypassed. The default value of the TryJetAuth attribute, equal to 1, causes Microsoft Access to attempt connection to the Oracle database using the Microsoft Access login information. You should set the `TryJetAuth` attribute equal to 0 in order to bypass the Microsoft Access login authentication. This step saves between one and two seconds when making the first connection to Oracle.

In Microsoft Access 2.0, you change the TryJetAuth attribute by modifying the msacc20.ini file, located in the Windows subdirectory. In Microsoft Access 95, Microsoft Access 97, and Microsoft Access 2000, you must modify the Windows registry using the Registry Editor (regedit). Table 5–1 displays the registry key location depending on the version of Microsoft Access you have installed on the system.

Table 5–1 Registry Key Location

Microsoft Access Release	Registry Key Location
Microsoft Access 95	`HKEY_LOCAL_MACHINE\Software\Microsoft\Jet\3.0\Engines\ODBC`
Microsoft Access 97	`HKEY_LOCAL_MACHINE\Software\Microsoft\Jet\3.5\Engines\ODBC`
Microsoft Access 2000	`HKEY_LOCAL_MACHINE\Software\Microsoft\Jet\4.0\Engines\ODBC`

Speeding Up Microsoft Jet database engine Operations

In order to ensure that the Microsoft Access forms and reports operate against the data residing in the Oracle database, the Migration Workbench generates ODBC link tables in the Microsoft Access database. These ODBC link tables reference the tables in the Oracle database. For more information, see the Modifying the Microsoft Access Database. Using ODBC link tables within an Microsoft Access application incurs a performance penalty. Specifically, although the connection time is faster due to caching of some ODBC data source information within the ODBC link table object, DML operations using ODBC link tables are slower.

The `CurrentDB` object is used extensively within Microsoft Access VBA code to reference database information. Prior to migration, the `CurrentDB` object references the local Microsoft Access tables. After the Migration Workbench has

modified the Microsoft Access database, the CurrentDB object references the ODBC link tables.

In order to speed up Microsoft Jet database engine operation with ODBC data sources, it is necessary to minimize the direct use of the ODBC link tables. One way of doing this within the VBA code is to use the Jet Workspace Database object instead of the CurrentDB object. The Database object does not reference the ODBC link tables. Instead, it requires a direct connection to the Oracle tables.

The modifications required within the VBA code are minimal as both the CurrentDB object and the Database object expose the same set of properties and methods.

To replace the CurrentDB object references with Database object references:

1. Create a global variable of type Database. For example:

```
Global dbOracle As Database
```

2. During the initialization stage, create the Database object by creating a connection to the Oracle database as follows:

```
Dim dsn As String
Dim uid As String
Dim pwd As String
Dim odbcConnectStr As String

dsn = "ora817"
uid = "system"
pwd = "manager"

’ build up the connect string
odbcConnectStr = "ODBC;DSN=" & dsn & ";UID=" & uid & ";PWD=" & pwd

' use Microsoft JET Workspace to make a connection to the Oracle database
Set dbOracle = DBEngine.Workspaces(0).OpenDatabase(dsn,_
dbDriverCompleteRequired, False, odbcConnectStr)
```

3. Replace all references to CurrentDB with a reference to dbOracle.

Note: Since the `Database` object references the Oracle tables directly, it is important to specify the Oracle table names instead of the Microsoft Access table names when you are performing table operations. The table names can be different between Microsoft Access and Oracle due to the different object naming restrictions imposed by both databases.

You cannot remove the ODBC link tables from Microsoft Access. This is because although the VBA code no longer references the ODBC link tables, the forms and reports within the database still directly reference these ODBC link tables. The effort involved in modifying the forms and reports so that they do not reference the ODBC link tables is considerable. It involves re-coding forms and reports as well as reprogramming all record navigation and manipulation. The Microsoft Jet database engine has not been bypassed as a result of these modifications. Because the Microsoft Jet database engine is still in use, a full table scan is still performed when retrieving records from the Oracle database. For more information on removing the Microsoft Jet database engine bottleneck, see the Eliminating the Microsoft Jet Database Engine topic.

Eliminating the Microsoft Jet Database Engine

The principal reason for the significant performance degradation of a Microsoft Access application using an ODBC data source is the Microsoft Jet database engine. The problem with Microsoft Jet database engine is that it always performs a full table scan when a table is queried. This means that when a Microsoft Access form or report references an Oracle table through an ODBC link the Microsoft Jet database engine must retrieve the entire contents of the table into local memory before it can perform the query. Eliminating Microsoft Jet database engine often results in an application that is faster than the original Microsoft Access application.

The alternative to using the Microsoft Jet database engine is to use ODBCDirect. ODBCDirect uses a `Connection` object that represents a pure connection to the destination Oracle database. When using ODBCDirect, the Microsoft Jet database engine is not loaded. When using ODBCDirect, all SQL statements are sent unaltered to the Oracle server for manipulation. These SQL statements are evaluated and interpreted on the Oracle server, reducing network traffic significantly. Only the subset of record information queried is sent back over the network to the Microsoft Access application.

Modifying the VBA code to use the ODBCDirect Connection object is not a straightforward procecdure. The property and method list of the `Connection` object is very different to those of the `CurrentDB` and `Database` objects. In addition, because the SQL statements are sent over the wire for manipulation by the Oracle server they must be in the correct Oracle syntax. Therefore, you should consider the following:

- Modify all SQL statements within the VBA code to comply with Oracle SQL syntax.
- Change SQL statements containing Microsoft Access specific expressions to Oracle syntax.
- Replace built-in Microsoft Access functions used within a SQL statement with the equivalent Oracle functions.

To replace the `CurrentDB or Database` object references with the `Connection` object references:

1. Create a global `ODBCDirect Connection` object reference as follows:

```
Global connOracle as Connection
```

2. Create an `ODBCDirect` work space as follows:

```
Dim DSN As String
Dim UID As String
Dim PWD AS String

DSN = "ora817"
UID = "system"
PWD = "manager"

wsODBC = DBEngine.CreateWorkspace(DSN, UID, PWD, dbUseODBC)
```

3. Create an `ODBCDirect Connection` object by connecting to the Oracle database as follows:

```
Dim ODBCconnectStr AS String

' build up the connect string
ODBCconnectStr = "ODBC;DSN=" & DSN & ";UID=" & UID & ";PWD=" & PWD

' open a connection to the Oracle database
Set connOracle = wsODBC.OpenConnection(DSN, dbDriverCompleteRequired, False,
ODBCconnectStr)
```

4. When manipulating data in the Oracle database, simply pass the SQL command directly to the Oracle server for processing as follows:

```
Dim sql As String
set sql = "select * from emp where empno > 10"
connOracle.execute sql
```

Qualified and Restricted Queries

You can reduce network traffic by requesting only the columns you need from a table. You should also use the most restrictive qualifications possible to reduce the size of the query result set.

Snapshots versus Dynasets

If possible, use Forward Only Snapshots to work with Oracle data, especially when the result set is small. For larger result sets and for queries that you must update, use a dynaset. Even if you are not going to update data, a dynaset is faster than a snapshot when the result set is large.

Comparing Use of Drop-Down Lists

Microsoft Access tries to minimize the amount of network traffic when it needs to populate a drop-down list option. When a snapshot is used to populate a drop-down list, Microsoft Access uses the same batch fetching of records that it uses to populate a grid or a form. Microsoft Access fetches an initial chunk of data (100 rows), then periodically retrieve sets of 100 rows from the server.

This process works smoothly unless you make an entry that does not match a row already fetched. In this case, Microsoft Access begins fetching records from the server until a match is found or until all records are retrieved. If the returned set is large, this step can be lengthy. It also may cause problems for you within the user interface.

Microsoft Access does not share queries for drop-down lists. A snapshot query is not reusable across multiple list boxes. Instead, Microsoft Access treats each activation of a query independently.

If a drop-down list is short, such as less than 100 records, it is probably sufficient to have Microsoft Access perform its normal operations. If the list is long, you may want to build a synchronized shadow table in Microsoft Access. Store the table

information locally in Microsoft Access and periodically synchronize the local table with information from Oracle.

A

Microsoft Access Error Messages

This appendix provides the complete list of error numbers and error messages returned by the Microsoft Jet database engine. It also indicates the class that each error belongs. This chapter includes the following sections:

- Error Message Classes
- Microsoft Jet Database Engine Error Messages

Error Message Classes

Table A–1 describes the classes that the Microsoft Jet database engine error messages belong:

Table A–1 Error Message Classes

Class	Description
BTRIEVE	BTRIEVE installable ISAM-specific errors (Microsoft Jet 2.5 and earlier).
DAO	DAO-specific errors.
DBASE	dBASE installable ISAM-specific errors.
DDL	Data Definition Language-specific errors.
EXCEL	Microsoft Excel installable ISAM-specific errors.
EXTENDED	Errors that may have extended error information.
IMEX	Generic import/export errors.
INST ISAM	Generic installable ISAM errors.
INTERNET	Internet-specific errors.

Table A–1 Error Message Classes

Class	Description
ISAM	Generic Microsoft Jet ISAM errors.
JPM	Microsoft Jet database engine errors related to property management.
MISC	Microsoft Jet database engine errors that don't fit into another category.
PARADOX	Paradaox installable ISAM-specific errors.
PARSE	Microsoft Jet database engine expression parsing errors.
QUERY	Microsoft Jet database engine errors related to queries.
REF INTEGRITY	Microsoft Jet database engine errors related to referential integrity.
REMOTE	Microsoft Jet database engine errors specific to ODBC.
REPLICATOR	Microsoft Jet database engine errors related to replication.
SECURITY	Microsoft Jet database engine errors related to security.
TEXT	Text installable ISAM-specific errors.
TLV	Table-level validation errors.
UNUSED	Microsoft Jet database engine errors that are no longer used or that have special meaning. Errors that have special meaning are usually translations from other errors and are not generated in the Microsoft Jet database engine code.

Microsoft Jet Database Engine Error Messages

Table A–2 lists the Microsoft Jet database engine database engine error messages. An asterisk (*) means that there is no Microsoft Jet error message text for a particular error message. An italicized item between angled brackets, such as <*name*>, represents a placeholder for a value that is given when the error message is displayed.

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
2420	Syntax error in number.	PARSE
2421	Syntax error in date.	PARSE
2422	Syntax error in string.	PARSE

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
2423	Invalid use of '.', '!', or '()'.	PARSE
2424	Unknown name.	PARSE
2425	Unknown function name.	PARSE
2426	Function isn't available in expressions.	PARSE
2427	Object has no value.	PARSE
2428	Invalid arguments used with domain function.	PARSE
2429	**In** Operator without ().	PARSE
2430	**Between** operator without **And**.	PARSE
2431	Syntax error (missing operator).	PARSE
2432	Syntax error (comma).	PARSE
2433	Syntax error.	PARSE
2434	Syntax error (missing operator).	PARSE
2435	Extra).	PARSE
2436	Missing),], or Item.	PARSE
2437	Invalid use of vertical bars.	PARSE
2438	Syntax error.	PARSE
2439	Wrong number of arguments with function.	PARSE
2440	**IIf** function without ().	PARSE
2442	Invalid use of parentheses.	PARSE
2443	Invalid use of **Is** operator.	PARSE
2445	Expression too complex.	PARSE
2446	Out of memory during calculation.	PARSE
2447	Invalid use of '.', '!', or '()'.	PARSE
2448	Can't set value.	PARSE
3000	Reserved error <*item*>; there is no message for this error.	UNUSED
3001	Invalid argument.	MISC
3002	Couldn't start session.	ISAM
3003	Couldn't start transaction; too many transactions already nested.	ISAM
3004*		UNUSED
3005	<*Database name*> isn't a valid database name.	ISAM

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3006	Database <*name*> is exclusively locked.	ISAM
3007	Can't open library database <*name*>.	ISAM
3008	The table <*name*> is already opened exclusively by another user, or it is already open through the user interface and cannot be manipulated programatically.	ISAM
3009	You tried to lock table <*name*> while opening it, but the table can't be locked because it is currently in use. Wait a moment, and then try the operation again.	ISAM
3010	Table <*name*> already exists.	MISC
3011	The Microsoft Jet database engine could not find the object <*name*>. Make sure the object exists and that you spell its name and the path name correctly.	MISC
3012	Object <*name*> already exists.	ISAM
3013	Couldn't rename installable ISAM file.	ISAM
3014	Can't open any more tables.	ISAM
3015	<*Index name*> isn't an index in this table. Look in the **Indexes** collection of the **TableDef** object to determine the valid index names.	ISAM
3016	Field won't fit in record.	ISAM
3017	The size of a field is too long.	MISC
3018	Couldn't find field <*name*>.	MISC
3019	Operation invalid without a current index.	ISAM
3020	You tried to call **Update** or **CancelUpdate** or attempted to update a **Field** in a recordset without first calling **AddNew** or **Edit**.	MISC
3021	No current record.	MISC
3022	The changes you requested to the table were not successful because they would create duplicate values in the index, primary key, or relationship. Change the data in the field or fields that contain duplicate data, remove the index, or redefine the index to permit duplicate values and try again.	ISAM
3023	**AddNew** or **Edit** already used.	QUERY
3024	Couldn't find file <*name*>.	MISC
3025	Can't open any more files.	ISAM
3026	Not enough space on disk.	ISAM
3027	Can't update. Database or object is read-only.	MISC

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3028	Can't start the application. The workgroup information file is missing or opened exclusively by another user.	ISAM
3029	Not a valid account name or password.	SECURITY
3030	<*Account name*> isn't a valid account name.	SECURITY
3031	Not a valid password.	SECURITY
3032	Can't perform this operation.	SECURITY
3033	You don't have the necessary permissions to use the <name> object. Have the system administrator of the person who created this object establish the appropriate permissions for you.	MISC
3034	You tried to commit or roll back a transaction without first using **BeginTrans**.	ISAM
3035*		
3036	Database has reached maximum size.	ISAM
3037	Can't open any more tables or queries.	MISC
3038*		
3039	Couldn't create index; too many indexes already defined.	ISAM
3040	Disk I/O error during read.	ISAM
3041	Can't open a database created with a previous version of the application.	ISAM
3042	Out of MS-DOS file handles.	ISAM
3043	Disk or network error.	UNUSED
3044	<*Path name*> isn't a valid path. Make sure that the path name is spelled correctly and that you are connected to the server where the file resides.	ISAM
3045	Couldn't use <*file name*>; file already in use.	ISAM
3046	Couldn't save; currently locked by another user.	ISAM
3047	Record is too large.	ISAM
3048	Can't open any more databases.	ISAM
3049	Can't open database <*name*>. It may not be a database that the application recognizes, or the file may be corrupt.	MISC
3050	Couldn't lock file.	ISAM
3051	The Microsoft Jet database engine cannot open the file <*name*>. It is already opened exclusively by another user, or you need permission to view its data.	MISC

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3052	MS-DOS file sharing lock count exceeded. You need to increase the number of locks installed with Share.exe.	ISAM
3053	Too many client tasks.	MISC
3054	Too many MEMO or OLE Object fields.	UNUSED
3055	Not a valid file name.	MISC
3056	Couldn't repair this database.	MISC
3057	Operation not supported on linked tables.	MISC
3058	Index or primary key can't contain a **NULL** value.	ISAM
3059	Operation canceled by user.	MISC
3060	Wrong data type for parameter <*parameter*>.	QUERY
3061	Too few parameters. Expected <*number*>.	EXTENDED
3062	Duplicate output alias <*name*>.	EXTENDED
3063	Duplicate output destination <*field name*>.	EXTENDED
3064	Can't open action query <*name*>.	QUERY
3065	Can't execute a select query.	QUERY
3066	Query must have at least one destination field.	EXTENDED
3067	Query input must contain at least one table or query.	EXTENDED
3068	Not a valid alias name.	QUERY
3069	The action query <*name*> cannot be used as a row source.	EXTENDED
3070	The Microsoft Jet database engine does not recognize <*name*> as a valid field name or expression.	QUERY
3071	This expression is typed incorrectly, or it is too complex to be evaluated. For example, a numeric expression may contain too many complicated elements. Try simplifying the expression by assigning parts of the expression to variables.	QUERY
3072	<*Visual Basic error string*>.	EXTENDED
3073	Operation must use an updateable query.	QUERY
3074	Can't repeat table name <*name*> in FROM clause.	EXTENDED
3075	<*Message*> in query expression <*expression*>.	EXTENDED
3076	<*Message*> in criteria expression.	EXTENDED
3077	<*Message*> in expression.	EXTENDED

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3078	The Microsoft Jet database engine cannot find the input table or query <*name*>. Make sure it exists and that its name is spelled correctly.	EXTENDED
3079	The specified field <*field name*> could refer to more than one table listed in the FROM clause of the SQL statement.	EXTENDED
3080	Joined table <*name*> not listed in FROM clause.	EXTENDED
3081	Can't join more than one table with the same name <*name*>.	EXTENDED
3082	JOIN operation <*operation*> refers to a field that isn't in one of the joined tables.	EXTENDED
3083	Can't use internal report query.	QUERY
3084	Can't insert data with action query.	QUERY
3085	Undefined function <*name*> in expression.	EXTENDED
3086	Couldn't delete from specified tables.	QUERY
3087	Too many expressions in GROUP BY clause.	QUERY
3088	Too many expressions in ORDER BY clause.	QUERY
3089	Too many expressions in DISTINCT clause.	QUERY
3090	Resultant table not allowed to have more than one AutoNumber field.	ISAM
3091	HAVING clause <*clause*> without grouping or aggregation.	UNUSED
3092	Can't use HAVING clause in TRANSFORM statement.	EXTENDED
3093	ORDER BY clause <*clause*> conflicts with DISTINCT.	EXTENDED
3094	ORDER BY clause <*clause*> conflicts with GROUP BY clause.	EXTENDED
3095	Can't have aggregate function in expression <*expression*>.	EXTENDED
3096	Can't have aggregate function in WHERE clause <*clause*>.	EXTENDED
3097	Can't have aggregate function in ORDER BY clause <*clause*>.	EXTENDED
3098	Can't have aggregate function in GROUP BY clause <*clause*>.	EXTENDED
3099	Can't have aggregate function in JOIN operation <*operation*>.	EXTENDED
3100	Can't set field <*name*> in join key to **Null**.	EXTENDED

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3101	The Microsoft Jet database engine can't find a record in the table <*table name*> with key matching field(s) <*field name*>.	EXTENDED
3102	Circular reference caused by <*query reference*>.	EXTENDED
3103	Circular reference caused by alias <*name*> in query definition's SELECT list.	EXTENDED
3104	Can't specify fixed column heading <*value*> in a crosstab query more than once.	EXTENDED
3105	Missing destination field name in SELECT INTO statement <*statement*>.	EXTENDED
3106	Missing destination field name in UPDATE statement <*statement*>.	EXTENDED
3107	Record(s) can't be added; no insert permission on <*name*>.	EXTENDED
3108	Record(s) can't be edited; no update permission on <*name*>.	EXTENDED
3109	Record(s) can't be deleted; no delete permission on <*name*>.	EXTENDED
3110	Couldn't read definitions; no read definitions permission for table or query <*name*>.	EXTENDED
3111	Couldn't create; no modify design permission for table or query <*name*>.	EXTENDED
3112	Record(s) can't be read; no read permission on <*name*>.	EXTENDED
3113	Can't update <*field name*>; field not updateable.	UNUSED
3114	Can't include MEMO or OLE Object when you select unique values <*statement*>.	EXTENDED
3115	Can't have MEMO or OLE Object fields in aggregate argument <*statement*>.	EXTENDED
3116	Can't have MEMO or OLE Object fields in criteria <*statement*> for aggregate function.	EXTENDED
3117	Can't sort on MEMO or OLE Object <*clause*>.	EXTENDED
3118	Can't join on MEMO or OLE Object <*name*>.	EXTENDED
3119	Can't group on MEMO or OLE Object <*clause*>.	EXTENDED
3120	Can't group on fields selected with '*' <*table name*>.	EXTENDED
3121	Can't group on fields selected with '*'.	EXTENDED
3122	You tried to execute a query that doesn't include the specified expression<*name*> as part of an aggregate function.	EXTENDED

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3123	Can't use '*' in crosstab query.	EXTENDED
3124	Can't input from internal report query <*name*>.	QUERY
3125	The database engine can't find <*name*>. Make sure it is a valid parameter or alisa name, that it doesn't include characters or punctuation, and that the name isn't too long.	MISC
3126	Invalid bracketing of name <*name*>.	EXTENDED
3127	The INSERT INTO statement contains the following unknown field name: <*field name*>. Make sure you've typed the name correctly, and try the operation again.	EXTENDED
3128	Specify the table containing the records you want to delete.	QUERY
3129	Invalid SQL statement; expected 'DELETE', 'INSERT', 'PROCEDURE', 'SELECT', or 'UPDATE'.	QUERY
3130	Syntax error in DELETE statement.	QUERY
3131	Syntax error in FROM clause.	QUERY
3132	Syntax error in GROUP BY clause.	QUERY
3133	Syntax error in HAVING clause.	QUERY
3134	Syntax error in INSERT INTO statement.	QUERY
3135	Syntax error in JOIN operation.	QUERY
3136	The LEVEL clause includes a reserved word or argument that is misspelled or missing, or the punctuation is incorrect.	QUERY
3137	Missing semicolon (;) at end of SQL statement.	QUERY
3138	Syntax error in ORDER BY clause.	QUERY
3139	Syntax error in PARAMETER clause.	QUERY
3140	Syntax error in PROCEDURE clause.	QUERY
3141	The SELECT statement includes a reserved word or an argument name that is misspelled or missing, or the punctuation is incorrect.	QUERY
3142	Characters found after end of SQL statement.	QUERY
3143	Syntax error in TRANSFORM statement.	QUERY
3144	Syntaxt error in UPDATE statement.	QUERY
3145	Syntax error in WHERE clause.	QUERY
3146	ODBC - call failed.	UNUSED

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3147	*	UNUSED
3148	*	UNUSED
3149	*	UNUSED
3150	*	UNUSED
3151	ODBC - connection to <*name*> failed.	EXTENDED
3152	*	UNUSED
3153	*	UNUSED
3154	ODBC - couldn't find DLL <*name*>.	REMOTE
3155	ODBC - insert on a linked table <*name*> failed.	EXTENDED
3156	ODBC - delete on a linked table <*name*> failed.	EXTENDED
3157	ODBC - update on a linked table <*name*> failed.	EXTENDED
3158	Couldn't save record; currently locked by another user.	INST ISAM
3159	Not a valid bookmark.	MISC
3160	Table isn't open.	INST ISAM
3161	Couldn't decrypt file.	INST ISAM
3162	You tried to assign the **Null** value to a variable that isn't a **Variant** data type.	MISC
3163	The field is too small to accept the amount of data you attempted to add. Try inserting or pasting less data.	MISC
3164	The field can't be updated because another user or process has locked the corresponding record or table.	MISC
3165	Couldn't open .inf file.	DBASE
3166	Cannot locate the requested Xbase memo file.	DBASE
3167	Record is deleted.	MISC
3168	Invalid .inf file.	DBASE
3169	The Microsoft Jet database engine could not execute the SQL statement because it contains a field that has an invalid data type.	QUERY
3170	Couldn't find installable ISAM.	UNUSED
3171	Couldn't find network path or user name.	PARADOX
3172	Couldn't open Paradox.net.	UNUSED
3173	Couldn't open table 'MSysAccounts' in the workgroup information file.	SECURITY

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3174	Couldn't open table 'MSysGroups' in the workgroup information file.	SECURITY
3175	Date is out of range or is in an invalid format.	INST ISAM
3176	Couldn't open file <*name*>.	IMEX
3177	Not a valid table name.	IMEX
3178	*	
3179	Encountered unexpected end of file.	IMEX
3180	Couldn't write to file <*name*>.	IMEX
3181	Invalid range.	IMEX
3182	Invalid file format.	IMEX
3183	Not enough space on temporary disk.	ISAM
3184	Couldn't execute query; couldn't find linked table.	EXTENDED
3185	SELECT INTO on a remote database tried to produce too many fields.	EXTENDED
3186	Couldn't save; currently locked by user <*name*> on machine <*name*>.	EXTENDED
3187	Couldn't read; currently locked by user <*name*> on machine <*name*>.	EXTENDED
3188	Couldn't update; currently locked by another session on this machine.	ISAM
3189	Table <*name*> is exclusively locked by user <*name*> on machine <*name*>.	UNUSED
3190	Too many fields defined.	ISAM
3191	Can't define field more than once.	ISAM
3192	Couldn't find output table <*name*>.	EXTENDED
3193	(unknown).	UNUSED
3194	(unknown).	UNUSED
3195	(expression).	UNUSED
3196	The database <*name*> is already in use by another person or process. When the database is available, try the operation again.	ISAM
3197	The Microsoft Jet database engine stopped the process because you and another user are attempting to change the same data at the same time.	MISC
3198	Couldn't start session. Too many sessions already active.	ISAM

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3199	Couldn't find reference.	REF INTEGRITY
3200	The record cannot be deleted or changed because table <*name*> includes related records.	EXTENDED
3201	You can't add or change a record because a related record is required in table <*name*>.	EXTENDED
3202	Couldn't save; currently locked by another user.	ISAM
3203	Subqueries cannot be used in the expression <*expression*>.	EXTENDED
3204	Database already exists.	ISAM
3205	Too many crosstab column headers <*value*>.	EXTENDED
3206	Can't create a relationship between a field and itself.	REF INTEGRITY
3207	Operation not supported on a Paradox table with no primary key.	PARADOX
3208	Invalid deleted setting in the Xbase key of the Windows registry.	DBASE
3209	*	UNUSED
3210	The connection string is too long.	QUERY
3211	The database engine couldn't lock table <*name*> because it's already in use by another person or process.	EXTENDED
3212	Couldn't lock table <*name*>; currently in use by user <*name*> on machine <*name*>.	UNUSED
3213	Invalid Date setting in the Xbase key of the Windows registry.	DBASE
3214	Invalid Mark setting in the Xbase key of the Windows registry.	DBASE
3215	Too many Btrieve tasks.	BTRIEVE
3216	Parameter <*name*> specified where a table name is required.	EXTENDED
3217	Parameter <*name*> specified where a database name is required.	EXTENDED
3218	Couldn't update; currently locked.	ISAM
3219	Invalid operation.	MISC
3220	Incorrect collating sequence.	PARADOX
3221	Invalid settings in the Btrieve key of the Windows registry.	BTRIEVE
3222	Query can't contain a Database parameter.	QUERY

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3223	*<Parameter name>* is invalid because it is too long or contains invalid characters.	EXTENDED
3224	Can't read Btrieve data dictionary.	BTRIEVE
3225	Encountered a record locking deadlock while performing a Btrieve operation.	BTRIEVE
3226	Errors encountered while using the Btrieve DLL.	BTRIEVE
3227	Invalid Century setting in the Xbase key of the Windows registry.	DBASE
3228	Invalid Collating Sequence setting in the Paradox key of the Windows registry.	PARADOX
3229	Btrieve - can't change field.	BTRIEVE
3230	Out-of-date Paradox lock file.	PARADOX
3231	ODBC - field would be too long; data truncated.	REMOTE
3232	ODBC - couldn't create table.	REMOTE
3233	*	UNUSED
3234	ODBC - remote query timeout expired.	REMOTE
3235	ODBC - data type not supported on server.	REMOTE
3236	*	UNUSED
3237	*	UNUSED
3238	ODBC - data out of range.	REMOTE
3239	Too many active users.	ISAM
3240	Btrieve - missing Btrieve engine.	BTRIEVE
3241	Btrieve - out of resources.	BTRIEVE
3242	Invalid reference in SELECT statement.	EXTENDED
3243	None of the import field names match fields in the appended table.	IMEX
3244	Can't import password-protected spreadsheet.	IMEX
3245	Couldn't parse field names from the first row of the import table.	IMEX
3246	Operation not supported in transactions.	MISC
3247	ODBC - linked table definition has changed.	REMOTE
3248	Invalid NetworkAccess setting in the Windows registry.	INST ISAM
3249	Invalid PageTimeout setting in the Windows registry.	INST ISAM

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3250	Couldn't build key.	ISAM
3251	Operation is not supported for this type of object.	MISC
3252	Can't open a form whose underlying query contains a user-defined function that attempts to set or get the form's **RecordsetClone** property.	MISC
3253	*	UNUSED
3254	ODBC - can't lock all records.	REMOTE
3255	*	UNUSED
3256	Index file not found.	DBASE
3257	Syntax error in WITH OWNERACCESS OPTION declaration.	QUERY
3258	The SQL statement couldn't be executed because it contains ambiguous outer joins. To force one of the joins to be performed first, create a separate query that performs the first join and then include that query in the SQL statement.	QUERY
3259	Invalid field data type.	MISC
3260	Couldn't update; currently locked by user <*name*> on machine <*name*>.	EXTENDED
3261	Table <*name*> is exclusively locked by user <*name*> on machine <*name*>.	EXTENDED
3262	Couldn't lock table <*name*>; currently in use by user <*name*> on machine <*name*>.	EXTENDED
3263	Invalid database object.	MISC
3264	No field defined - cannot append **TableDef** or **Index**.	DAO
3265	Item not found in this collection.	DAO
3266	Can't append a **Field** that is already a part of a **Fields** collection.	DAO
3267	Property can be set only when the **Field** is part of a **Recordset** object's **Fields** collection.	DAO
3268	Can't set this property once the object is part of a collection.	DAO
3269	Can't append an **Index** that is already part of an **Indexes** collection.	DAO
3270	Property not found.	DAO
3271	Invalid property value.	DAO
3272	Object isn't a collection.	DAO

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3273	Method not applicable for this object.	DAO
3274	External table isn't in the expected format.	INST ISAM
3275	Unexpected error from the external database driver *<error number>*.	INST ISAM
3276	Invalid database object reference.	MISC
3277	Can't have more than 10 fields in an index.	ISAM
3278	The Microsoft Jet database engine has not been initialized.	MISC
3279	The Microsoft Jet database engine has already been initialized.	MISC
3280	Can't delete a field that is part of an index or is need by the system.	ISAM
3281	Can't delete this index or table. It is either the current index or is used in a relationship.	ISAM
3282	Operation is not supported on a table that contains data.	ISAM
3283	Primary key already exists.	ISAM
3284	Index already exists.	ISAM
3285	Invalid index definition.	ISAM
3286	Format of memo file doesn't match specified external database format.	INST ISAM
3287	Can't create index on the given field.	ISAM
3288	Paradox index is not primary.	PARADOX
3289	Syntax error in CONSTRAINT clause.	DDL
3290	Syntax error in CREATE TABLE statement.	DDL
3291	Syntax error in CREATE INDEX statement.	DDL
3292	Syntax error in field definition.	DDL
3293	Syntax error in ALTER TABLE statement.	DDL
3294	Syntax error in DROP INDEX statement.	DDL
3295	Syntax error in DROP TABLE or DROP INDEX.	DDL
3296	Join expression not supported.	MISC
3297	Couldn't import table or query. No records found, or all records contain errors.	IMEX
3298	There are several tables with than name. Please specify owner in the format 'owner.table'.	REMOTE

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3299	ODBC Specification Conformance Error <*message*>. Report this error to the developer of the application.	UNUSED
3300	Can't create a relationship.	REF INTEGRITY
3301	Can't perform this operation; features in this version are not available in databases with older formats.	MISC
3302	Can't change a rule while the rules for this table are in use.	TLV
3303	Can't delete this field. It's part of one or more relationships.	REF INTEGRITY
3304	You must enter a personal identifier (PID) consisting of at least four and no more than 20 characters and digits.	SECURITY
3305	Invalid connection string in pass-through query.	REMOTE
3306	You've written a subquery that can return more than one field without using the EXISTS reserved word in the main query's FROM clause. Revise the SELECT statement of the subquery to request only one field.	QUERY
3307	The number of columns in the two selected tables or queries of a union query don't match.	QUERY
3308	Invalid TOP argument in select query.	EXTENDED
3309	Property setting can't be larger than 2K.	JPM
3310	This property isn't supported for external data sources for for databases created with a previous version of Microsoft Jet.	JPM
3311	Property specified already exists.	JPM
3312	Validation rules and default values can't be placed on system or linked tables.	TLV
3313	Can't place this validation expression on this field.	TLV
3314	The field <*name*> can't contain a **Null** value because the Required property for this field is set to **True**. Enter a value in this field.	EXTENDED
3315	Field <*name*> can't be a zero-lenght string.	EXTENDED
3316	<*Table-level validation text*>.	EXTENDED
3317	One or more values are prohibited by the validation rule <*rule*> set for <*field name*>. Enter a value that the expression for this field can accept.	UNUSED
3318	Values specified in a TOP clause are not allowed in delete queries or reports.	EXTENDED
3319	Syntax error in union query.	QUERY

Table A–2 Microsoft Jet Database Engine Error Messages

Error Number	Error Message	Class
3320	<*Error*> in table-level validation expression.	EXTENDED
3321	No database specified in connection string or **IN** clause.	REMOTE
3322	Crosstab query contains one or more invalid fixed column headings.	EXTENDED
3323	The query cannot be used as a row source.	QUERY
3324	The query is a DDL query and cannot be used as a row source.	QUERY
3325	Pass-through query with **ReturnsRecords** property set to **True** did not return any records.	REMOTE
3326	This **Recordset** is not updateable.	EXTENDED
3327	Field <*name*> is based on an expression and can't be edited.	EXTENDED
3328	Table <*name*> is read-only.	EXTENDED
3329	Record in table <*name*> was deleted by another user.	EXTENDED
3330	Record in table <*name*> is locked by another user.	EXTENDED
3331	To make changes to this field, first save the record.	EXTENDED
3332	Can't enter value into blank field on 'one' side of outer join.	EXTENDED
3333	Records in table <*name*> would have no record on the 'one' side.	EXTENDED
3334	Can be present only in version 1.0 format.	ISAM
3335	DeleteOnly called with non-zero cbData.	JPM

B

Code and Query Samples

This appendix contains samples that support emulation of the AUTONUMBER data type, CASCADE UPDATE referential integrity, and name mapping queries. It contains the following sections:

AUTONUMBER Data Type Emulation

Microsoft Access supports a AUTONUMBER data type. The AUTONUMBER data type provides a monotonically increasing sequence of long integers for a column in a native Jet DBMS file. Oracle supports sequences. Sequences generate a set of numbers that can be used in columns as unique identifiers. An important difference between Microsoft Access AUTONUMBERs and Oracle sequences is that a trigger is required in Oracle in order to place a sequence number in a column when a new record is inserted into a table.

When Jet has an ODBC link to an Oracle table and an Oracle trigger changes or initializes the key values at the time of an insert, Jet performs a sequence of queries to retrieve the new key value so that the inserted row can become a member of the dynaset. If Jet has trouble re-selecting the inserted row, the rows appears as #DELETED to the user. This does not apply to updates.

The example below shows how to emulate a AUTONUMBER data type in Oracle.

Oracle Table

```
CREATE TABLE OTBLAUTONUMBERTEST(
PK NUMBER(10,0),
NAME VARCHAR2(50),
CONSTRAINT PK_OTBLAUTONUMBERTEST PRIMARY KEY (PK))
```

Oracle Sequence

```
CREATE SEQUENCE   TEST     INCREMENT BY 1    START WITH 1000
```

Trigger Code

```
Create Trigger TRG_CNT_OTBLAUTONUMBERTEST
Before INSERT OR UPDATE on OTBLAUTONUMBERTEST
FOR EACH ROW
DECLARE
   iAUTONUMBER SCOTT.OTBLAUTONUMBERTEST.PRIMARYKEY%TYPE;
   cannot_change_AUTONUMBER EXCEPTION;

BEGIN
   IF INSERTING THEN
      SELECT TEST.NEXTVAL into iAUTONUMBER FROM dual;
      :new.PRIMARYKEY := iAUTONUMBER;
   END IF; -- End of Inserting Code

   IF UPDATING THEN
      -- Do not allow the PK to be changed.
      IF NOT(:new.PRIMARYKEY = :old.PRIMARYKEY) THEN
         RAISE cannot_change_AUTONUMBER;
      END IF;

   END IF; -- End of Updating Code

EXCEPTION
   WHEN cannot_change_AUTONUMBER THEN
      raise_application_error(-20000,'Cannot Change AUTONUMBER Value');
END;
```

This trigger emulates the AUTONUMBER data type by trapping both INSERT and UPDATE operations on a table. On any insert the trigger obtains the next value in the sequence TEST for the PRIMARYKEY column. On UPDATEs, the trigger checks to see if the user is trying to update the AUTONUMBER. If yes, an exception is raised and the error is passed back to Microsoft Access.

It is not recommended to silently protect the AUTONUMBER on UPDATE. In the following example, Jet is unable to successfully manage the dynaset and produces unpredictable results:

```
IF UPDATING THEN
   -- Do not allow the PK to be changed.

   IF NOT(:new.PRIMARYKEY = :old.PRIMARYKEY) THEN
      :new.PRIMARYKEY := :old.PRIMARYKEY);
   END IF;

END IF; -- End of Updating Code
```

You can restrict AUTONUMBER field emulation by adding code in the trigger to allow Microsoft Access to pass a value for the AUTONUMBER on a row insert. The code below generates a new AUTONUMBER value only if the passed value is NULL.

```
IF INSERTING THEN
   IF (:new.PRIMARYKEY IS NULL) THEN
      SELECT test.NEXTVAL into iAUTONUMBER FROM dual;
      :new.PRIMARYKEY := iAUTONUMBER;
   END IF:
END IF; -- End of Inserting Code
```

Name Mapping Query

To begin building a name mapping query in Microsoft Access, use either the QBE or SQL window to define the query. In this example, the original Microsoft Access table is called SeqDateTable and is exported to Oracle as O_SEQDATETABLE. After the export, the table is attached to Jet as R_SeqDateTable.

When the following query is saved as SeqDateTable, it takes the place of the original table and complete the mapping to Oracle. The query maps the column names PRIMARYKEY, O_SEQUENCE and FIRSTDATE to PrimaryKey, Sequence and FirstDate for use by Microsoft Access.

```
SELECT NameMapper.PRIMARYKEY AS PrimaryKey,
NameMapper.O_SEQUENCE AS Sequence,
NameMapper.FIRSTDATE AS FirstDate
FROM R_SEQDATETABLE;
```

Default Values

Oracle supports declarative default values. However, when moving an application from Microsoft Access to Oracle, you may encounter situations where you need an insert trigger to support defaults. A reasonable design decision is to move all default processing to triggers to centralize the code and reduce maintenance complexity. The following code sample demonstrates supporting default values in a trigger:

```
CREATE OR REPLACE TRIGGER BIU_M2
 BEFORE INSERT OR UPDATE
 ON M2
 FOR EACH ROW

   BEGIN
      IF INSERTING THEN
      /* Manage Default Values if a new value is NULL */
         IF :new.Address IS NULL THEN
            :new.Address := 'Default';
         END IF;
      END IF; -- End of Inserting Code
   END; -- Trigger BI_M2
```

Column and Table Validation

Oracle supports CHECK statements that you can use to enforce table constraints and column constraints. However, when moving an application from Microsoft Access to Oracle, you may encounter situations where you need an insert trigger to support validation. The code sample below demonstrates supporting validation in a trigger. The <*Access Validation Code*> indicates where you can insert the validation code from a Microsoft Access application.

```
CREATE OR REPLACE TRIGGER BIU_M2
 BEFORE INSERT OR UPDATE
 ON M2
 FOR EACH ROW

 BEGIN
      -- Validation Code
      IF NOT ( <Access Validation Code > ) THEN
         raise_application_error (-20000, '<Access Error Message>');
      END IF;
END; -- Trigger BI_M2
```

CASCADE UPDATE Trigger Code

Oracle does not provide direct support for CASCADE UPDATE referential integrity constraints. CASCADE UPDATE support means that when a primary key is changed, that change is made to all associated foreign keys in linked tables. CASCADE UPDATE is not a common design feature in applications. Primary keys should be stable, usually for the life of an application.

The following code example is based on two tables:

```
create table M1 (
f1 number,
f2 number,
f3 number )
create table M2 (f1 number,
f2 number,
f3 number )
alter table M1 add primary key (f1)
alter table M2 add primary key (f1)
```

This definition supports one-to-many cardinality. To add support for one-to-one cardinality add the following:

```
alter table M1 add constraint uq_M1_001 unique (f2, f3)
alter table M2 add constraint uq_M2_001 unique (f2, f3)
```

The following code implements CASCADE UPDATE code for the two tables, M1 and M2. This example uses two columns in the primary / foreign key relationships. This relationship is more complex than most and is used to fully illustrate the proper code.

Declarative and procedural support for referential integrity cannot coexist between two tables. To support CASCADE UPDATE between two tables, all declarative primary / foreign key relationships and referential integrity between the tables must be removed and supported instead with procedural code. This is outlined in the following code sample:

```
CREATE OR REPLACE PACKAGE P_M1 AS
    fire_trigger boolean := TRUE;
 END P_M1;

 CREATE OR REPLACE PACKAGE P_M2 AS
    fire_trigger boolean := TRUE;
```

```
 END P_M2;

 CREATE OR REPLACE PACKAGE UQ_M1_M2 AS

 PROCEDURE cascade_update (
           o_F2    IN  number,
           o_F3    IN  number,
           n_F2    IN  number,
           n_F3    IN  number,
           bResult OUT boolean );

       PROCEDURE cascade_delete (
          F2      IN  number,
          F3      IN  number,
          bResult OUT boolean );

       FUNCTION pk_exists (
          F2      IN  number,
          F3      IN  number) RETURN boolean;

       FUNCTION fk_exists (
          F2      IN  number,
          F3      IN  number) RETURN boolean;

 END  UQ_M1_M2;

CREATE OR REPLACE PACKAGE BODY UQ_M1_M2 AS

   /* Procedure cascade_update is called when field(s) */
   /*   F2 or                                          */
   /*   F3                                             */
   /*   are changed in table M1.                       */
   /*   The changes are cascaded in table M2           */

   PROCEDURE cascade_update (
             o_F2    IN  number,
             o_F3    IN  number,
             n_F2    IN  number,
             n_F3    IN  number,
             bResult OUT boolean ) IS

   CURSOR d_cur (n1 number, n2 number) IS
      SELECT * FROM m2
      WHERE f2 = n1 AND f3 = n2
      FOR UPDATE of f2, f3;
```

```
BEGIN

   FOR d_cur_rec IN d_cur ( o_F2, o_F3 )
   LOOP
      UPDATE M2 SET f2 = n_F2, f3 = n_F3
         WHERE CURRENT OF d_cur;
   END LOOP; -- Detail Record Loop

   bResult := true;

END cascade_update;

/* Procedure cascade_delete is called when a record */
/*   in M1 is being deleted and associated          */
/*   child records in M2 must also be deleted.      */

PROCEDURE cascade_delete (
        F2      IN  number,
        F3      IN  number,
        bResult OUT boolean ) IS

CURSOR d_cur (n1 number, n2 number) IS
   SELECT * FROM m2
   WHERE f2 = n1 AND f3 = n2
   FOR UPDATE;

BEGIN

   FOR d_cur_rec IN d_cur ( F2, F3 )
   LOOP
      DELETE FROM M2
         WHERE CURRENT OF d_cur;
   END LOOP; -- Detail Record Loop

   bResult := true;

END cascade_delete;

/* Procedure pk_exists is called to determine is a given
   primary key exists in table M1                          */

FUNCTION pk_exists (
     F2      IN  number,
     F3      IN  number) RETURN boolean IS
```

```
l_F2        number;
l_F3        number;
bResult     boolean;

CURSOR p_cur (n1 number, n2 number) IS
   SELECT F2, F3 FROM m1
   WHERE f2 = n1 AND f3 = n2;

BEGIN
   OPEN p_cur( F2, F3 );
   FETCH p_cur INTO l_F2, l_F3;
   IF p_cur%NOTFOUND THEN
      bResult := false;
   ELSE
      bResult := true;
   END IF;

   CLOSE p_cur;

   RETURN( bResult );

END pk_exists;

/* Procedure pk_exists is called to determine is a given
   primary key exists in table M1                          */

FUNCTION fk_exists (
     F2      IN  number,
     F3      IN  number) RETURN boolean IS

l_F2         number;
l_F3         number;
bResult      boolean;

CURSOR d_cur (n1 number, n2 number) IS
   SELECT F2, F3 FROM m2
   WHERE f2 = n1 AND f3 = n2;

BEGIN
   OPEN d_cur( F2, F3 );
   FETCH d_cur INTO l_F2, l_F3;
   IF d_cur%NOTFOUND THEN
      bResult := false;
   ELSE
```

```
        bResult := true;
      END IF;

      CLOSE d_cur;

      RETURN( bResult );

  END fk_exists;

END UQ_M1_M2;


CREATE OR REPLACE TRIGGER AUD_M1
AFTER UPDATE OR DELETE
ON M1
FOR EACH ROW

  DECLARE
  bResult_OK      BOOLEAN;
  bCascadeDeletes BOOLEAN  := TRUE;

  BEGIN

     IF UPDATING THEN
        IF (:old.F2 <> :new.F2) OR (:old.F3 <> :new.F3) THEN
           P_M2.fire_trigger := FALSE;
           UQ_M1_M2.cascade_update( :old.F2, :old.F3, :new.F2, :new.F3,
                                    bResult_OK );
           P_M2.fire_trigger := TRUE;
        END IF;
     END IF; -- End of Updating Code

     IF DELETING THEN
        IF bCascadeDeletes THEN
           UQ_M1_M2.cascade_delete( :old.F2, :old.F3, bResult_OK );
        ELSE
           IF UQ_M1_M2.fk_exists( :old.F2, :old.F3 ) THEN
              raise_application_error( -20000, 'Rows exist in child table');
           END IF;
        END IF;
     END IF; -- End of Deleting Code

  END; -- Trigger AUD_M1
```

```
CREATE OR REPLACE TRIGGER AIU_M2
AFTER INSERT OR UPDATE
ON M2
FOR EACH ROW

  DECLARE
  bResult_OK    BOOLEAN;

  BEGIN

     IF INSERTING THEN
        IF NOT( UQ_M1_M2.pk_exists( :new.F2, :new.F3 ) ) THEN
           raise_application_error (-20000, 'No corresponding row in parent
                      table');
        END IF;
     END IF; -- End of Inserting Code

     IF ( UPDATING AND P_M2.fire_trigger ) THEN
        IF NOT( UQ_M1_M2.pk_exists( :new.F2, :new.F3 ) ) THEN
           raise_application_error (-20000, 'No corresponding row in parent
table');
        END IF;
     END IF; -- End of Updating Code

  END; -- Trigger AUD_M2
```

Index

N

O

P

Q

R

S

T

V