Oracle® Database Gateway for Adabas

User's Guide

11*g* Release 2 (11.2)

E12074-01

July 2009

ORACLE®

Oracle Database Gateway for Adabas User's Guide, 11*g* Release 2 (11.2)

E12074-01

Copyright © 2008, 2009, Oracle and/or its affiliates. All rights reserved.

Primary Author: Jeanne Wiegelmann

Contributing Author: Maitreyee Chaliha, Sami Zeitoun, Oussama Mkaabal

Contributor: Vira Goorah, Peter Wong

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Preface vii

Audience vii
Documentation Accessibility vii
Related Documents viii
Conventions viii

1 Getting Started with Oracle Database Gateways

Overview 1-1
Oracle Heterogeneous Services 1-1
Oracle Database Gateways 1-2
Gateway Architecture 1-3
Gateway Process Flow 1-4
Setup Flow 1-5

2 Gateway Features and Restrictions

Accessing Adabas Data 2-1
Exporting Predict Metadata into Adabas ADD 2-1
SQL Support 2-2
Pass-Through Feature 2-3
Naming Rules 2-3
Rules for Naming Objects 2-3
Case Sensitivity 2-3
Data Types 2-3
Binary Literal Notation 2-4
Data Type Conversion 2-4
SQL Execution 2-4
Row Selection 2-4
Empty Strings 2-4
Empty Bind Variables 2-5
Locking 2-5
Handling Non-Relational Data 2-5
Restrictions 2-5
SQL Restrictions 2-6
Unsupported SQL Functions 2-6
SQL Syntax 2-6

Transaction Capability 2-7
COMMIT or ROLLBACK in PL/SQL Cursor Loops 2-7
Pass-Through Feature 2-8
Database Links 2-8
Data Types 2-8
Encrypted Format Login 2-8
Data Dictionary Views and PL/SQL 2-8
Adabas Restrictions 2-8

3 Normalizing Non-Relational Data

Overview of Normalizing Non-Relational Data 3-1
About Arrays 3-1
Representing Metadata 3-1
Methods of Handling Arrays 3-5
Virtual Views 3-5
Sequential Flattening (Bulk Load of Array Data) 3-7

4 Setting the Array Handling Policy

Setting the Default Behavior for Array Handling 4-1
Modifying the Array Handling Policy for a Specific Table 4-2

A Data Type Conversion

Conversion of Predict Metadata A-1
Conversion of Metadata from DDM Files A-2

B Supported SQL Syntax and Functions

Supported SQL Statements B-1
DELETE B-1
INSERT B-1
SELECT B-1
UPDATE B-2
Oracle Functions B-2
Functions Not Supported by Adabas B-2
Functions Supported by Adabas B-2
Arithmetic Operators B-2
Comparison Operators B-3
Group Functions B-3
String Functions B-3
Other Functions B-3

C Data Dictionary

Data Dictionary Support C-1
Accessing the Gateway Data Dictionary C-1
Supported Views and Tables C-2
Default Column Values C-3

Adabas Data Dictionary Descriptions C-3

D Globalization Support

Character Set Terminology D-1
Globalization Support Settings D-2

Index

Preface

This manual describes the Oracle Database Gateway for Adabas, which enables Oracle client applications to access Adabas data through Structured Query Language (SQL). The gateway, with the Oracle Database, creates the appearance that all data resides on a local Oracle Database, even though the data can be widely distributed.

Audience

This manual is intended for Oracle database administrators who perform the following tasks:

- Installing and configuring the Oracle Database Gateway for Adabas
- Diagnosing gateway errors
- Using the gateway to access Adabas data

> **Note:** You should understand the fundamentals of Oracle Database Gateways before using this guide to install or administer the gateway.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. 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. Accessibility standards will continue to evolve over time, and Oracle 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 more information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation

Screen readers 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, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at `http://www.fcc.gov/cgb/consumerfacts/trs.html`, and a list of phone numbers is available at `http://www.fcc.gov/cgb/dro/trsphonebk.html`.

Related Documents

For more information, see the following documents in the Oracle Database Gateway 11*g* Release 2 (11.2) documentation set:

- *Oracle Database New Features*
- *Oracle Call Interface Programmer's Guide*
- *Oracle Enterprise Manager Administrator's Guide*
- *Oracle Database Administrator's Guide*
- *Oracle Application Developer's Guide - Fundamentals*
- *Oracle Database Concepts*
- *Oracle Database Performance Tuning Guide and Reference*
- Oracle Database Performance Planning
- *Oracle Database Error Messages*
- *Oracle Database Globalization Support Guide*
- *Oracle Database Reference*
- *Oracle SQL Reference*
- *Oracle Net Services Administrator's Guide*
- *SQL*Plus User's Guide and Reference*
- *Oracle Database Heterogeneous Connectivity User's Guide*
- *Oracle 2 Day DBA*
- *Oracle Database Security Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.

Convention	Meaning
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Getting Started with Oracle Database Gateways

This chapter contains the following sections:

- Overview
- Oracle Heterogeneous Services
- Oracle Database Gateways
- Gateway Architecture
- Gateway Process Flow
- Setup Flow

Overview

Heterogeneous data access is a problem that affects a lot of companies. A lot of companies run several different database systems. Each of these systems stores data and has a set of applications that run against it. Consolidation of this data in one database system is often hard—in large part because many of the applications that run against one database may not have an equivalent that runs against another. Until such time as migration to one consolidated database system is made feasible, it is necessary for the various heterogeneous database systems to interoperate.

Oracle Database Gateways provide the ability to transparently access data located in a non-Oracle system from an Oracle environment. This transparency eliminates the need for application developers to customize their applications to access data from different non-Oracle systems, thus decreasing development efforts and increasing the mobility of the application. Applications can be developed using a consistent Oracle interface for both Oracle and Adabas.

Gateway technology is composed of two parts: a component that has the generic technology to connect to a non-Oracle system, which is common to all the non-Oracle systems, called Heterogeneous Services, and a component that is specific to the non-Oracle system that the gateway connects to. Heterogeneous Services, in conjunction with the Database Gateway agent and Oracle Connect for IMS, VSAM, and Adabas Gateways, enables transparent access to non-Oracle systems from an Oracle environment.

Oracle Heterogeneous Services

Heterogeneous Services provides the generic technology for connecting to non-Oracle systems. As an integrated component of the database, Heterogeneous Services can

exploit features of the database, such as the powerful SQL parsing and distributed optimization capabilities.

Heterogeneous Services extend the Oracle SQL engine to recognize the SQL and procedural capabilities of the remote non-Oracle system and the mappings required to obtain necessary data dictionary information. Heterogeneous Services provides two types of translations: the ability to translate Oracle SQL into the proper dialect of the non-Oracle system and the ability to handle data dictionary translations so that the metadata of the non-Oracle system is displayed in the local format. For situations where no translations are available, native SQL can be issued to the non-Oracle system using the pass-through feature of Heterogeneous Services.

Heterogeneous Services also maintains the transaction coordination between Oracle and the remote non-Oracle system, such as providing the global transaction protocol to ensure distributed transaction integrity, even for non-Oracle systems that do not natively support global transactions.

> **See Also:** *Oracle Database Heterogeneous Connectivity User's Guide* for more information about Heterogeneous Services.

Oracle Database Gateways

Oracle Database Gateway for Adabas allows Oracle client applications to access Adabas data through Structured Query Language (SQL). The gateway, with the Oracle database server, creates the appearance that all data resides on a local Oracle database server, even though data might be widely distributed. If data is moved from this data source to an Oracle database, no changes in the client application's design or function are needed because the gateway handles all differences in data types or SQL functions between the applications and the database.

Using Oracle SQL, Oracle client applications can access the Adabas data source as if the data was stored in an Oracle table. A single SQL statement can access data residing in Oracle and Adabas data sources, performing heterogeneous joins and subselects. This means that you can develop one set of portable applications to use against Oracle and these non-relational data sources. You can continue to develop new information systems without losing your investment in existing data and applications.

Transactions updating Oracle and this non-relational data source are automatically protected by the Oracle global transactions feature. Use of synonyms is another Oracle feature. By setting up synonyms in the Oracle database server that point to database links to Adabas files, the physical location of the data is transparent to the client application. This allows future migration of data from Adabas to Oracle to be transparent to the client applications.

The gateway requires the Oracle database server, Oracle Connect for IMS, VSAM, and Adabas Gateways, and Oracle Studio for IMS, VSAM, and Adabas Gateways. All other Oracle products are optional. However, using other Oracle products with the gateway can greatly extend the gateway's capabilities.

The gateway can be installed on a computer where the Oracle database is installed, or on a second, standalone machine. Each configuration has its advantages and disadvantages. The issues to consider when you determine where to install the gateway are network traffic, availability of the operating system platform, hardware resources, and storage.

Oracle Connect for IMS, VSAM, and Adabas Gateways must be installed on the z/OS system where the Adabas data source is installed. To be able to configure and manage Oracle Connect for IMS, VSAM, and Adabas Gateways, you need to install Oracle

Studio for IMS, VSAM, and Adabas Gateways on a computer running Windows or Linux.

Gateway Architecture

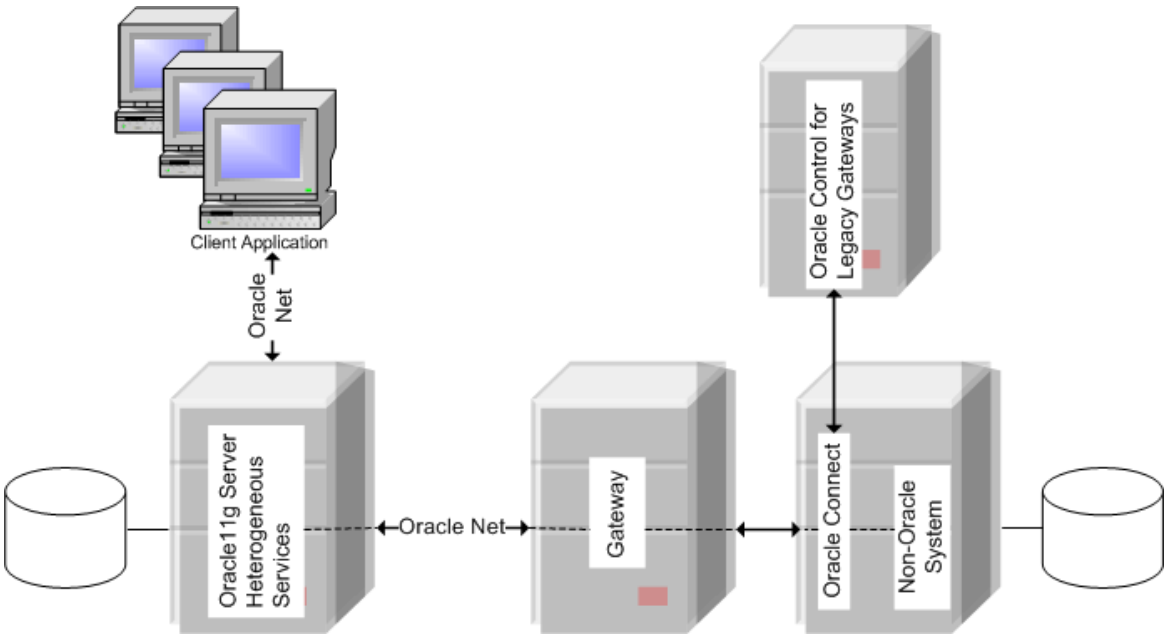
The gateway is invoked by the listener. The gateway is not multi-threaded and cannot support shared database links. Each gateway session spawns a separate gateway process, and connections cannot be shared.

The gateway is located on a Windows or UNIX computer. The non-relational data source resides on a computer running IBM z/OS. The Oracle database server can reside on the same machine as the gateway or on another machine.

The gateway interacts with the Oracle database server to interface between client applications and the Adabas data source, as shown in Figure 1–1.

Note: The non-Oracle system in Figure 1–1 and Figure 1–2 represents the Adabas data source.

Figure 1–1 Gateway Processing



- Client applications, such as Developer, connect to the Oracle database server by using Oracle Net.
- The Oracle database server, which includes Heterogeneous Services and the database itself, resides on a single system. This Oracle database server also stores definitions of database links for the non-Oracle system.
- The gateway to Adabas resides on a second system.
- The Adabas data source, together with Oracle Connect for IMS, VSAM, and Adabas Gateways, resides on a third system, which is an IBM z/OS platform.
- Oracle Studio for IMS, VSAM, and Adabas Gateways, which is used to configure Oracle Connect for IMS, VSAM, and Adabas Gateways, resides on a fourth system, which can be running a Windows or Linux operating system.

- The Oracle database server on the first system uses Oracle Net to connect directly to the Adabas gateway on the second system. The gateway itself uses the Oracle Connect for IMS, VSAM, and Adabas Gateways protocol to connect to Oracle Connect for IMS, VSAM, and Adabas Gateways, which then connects to the non-Oracle system using the same protocol.

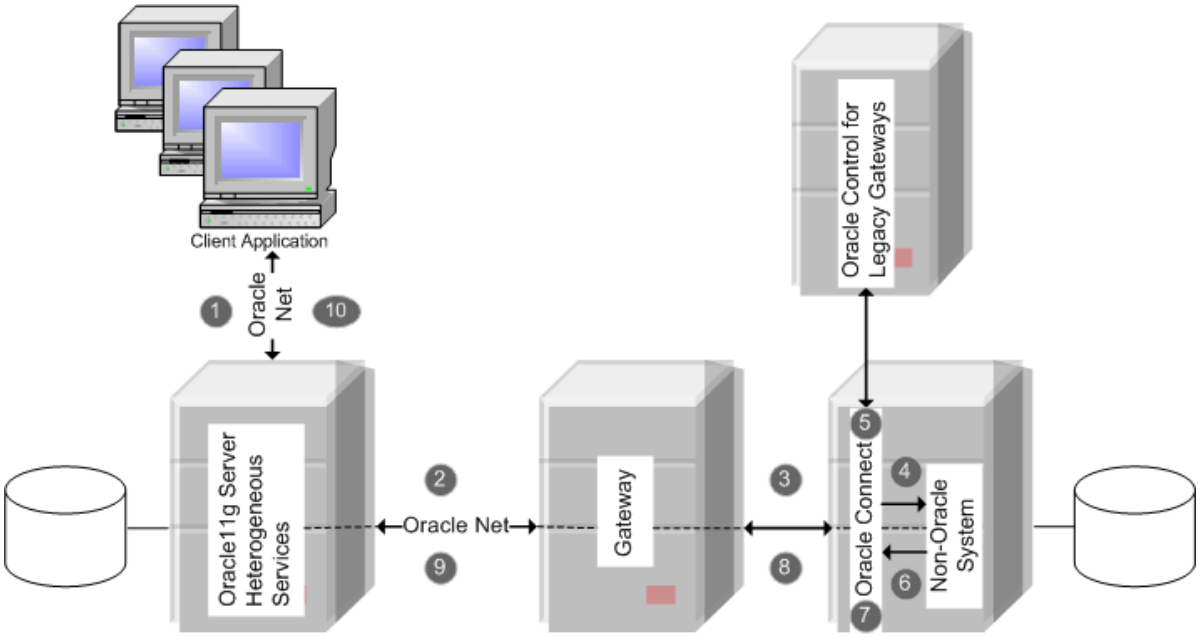
The Oracle database server and the gateway work together to present the appearance of a single Oracle database to the client. All data accessed by the client appears to reside in a single Oracle database. The client application sends a request to the Oracle database server, and the Oracle database server sends the request to the gateway.

For the first transaction in a session, the gateway logs into the Adabas data source using a username and password that is valid in the respective data source. The gateway converts the SQL statement to a native Adabas statement, and the Adabas data source performs the request. The gateway converts the retrieved data to a format compatible with the Oracle database server and returns the results to the Oracle database server, which returns the results to the client application.

Gateway Process Flow

Figure 1–2 shows a typical gateway process flow. The steps explain the sequence of the events that occurs when a client application queries the Adabas data source through the gateway.

Figure 1–2 Gateway Process Flow



1. The client application sends a query over Oracle Net to the Oracle database server.
2. The Oracle database server sends the query over to the gateway, again using Oracle Net.
3. The gateway passes the query on to Oracle Connect for IMS, VSAM, and Adabas Gateways.
4. For the first transaction in a session, Oracle Connect for IMS, VSAM, and Adabas Gateways logs into the Adabas data source using a user name and password that is valid in the respective data source.

5. Oracle Connect for IMS, VSAM, and Adabas Gateways converts the Oracle SQL statement into a data access operation understood by the data source.
6. Oracle Connect for IMS, VSAM, and Adabas Gateways retrieves the data.
7. Oracle Connect for IMS, VSAM, and Adabas Gateways converts the retrieved data into a format compatible with the Oracle database server.
8. Oracle Connect for IMS, VSAM, and Adabas Gateways passes the data to the gateway using the Oracle Connect for IMS, VSAM, and Adabas Gateways protocol.
9. The gateway returns the query results to the Oracle database server, again using Oracle Net.
10. The Oracle database server passes the query results to the client application by using Oracle Net. The database link remains open until the gateway session is finished or the database link is explicitly closed.

Setup Flow

To be able to access Adabas data, you need to perform the tasks described in the following list, in the specified order. Each step in the list directs you to the relevant manual or chapter.

1. Install Oracle Connect for IMS, VSAM, and Adabas Gateways

   See: *Oracle Connect for IMS, VSAM, and Adabas Gateways Installation and Configuration Guide for IBM z/OS* for information on installing Oracle Connect for IMS, VSAM, and Adabas Gateways

2. Install Oracle Studio for IMS, VSAM, and Adabas Gateways

   See: *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64*

3. Configure Oracle Connect for IMS, VSAM, and Adabas Gateways

   See: *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64*

4. Set up the connection to Oracle Connect for IMS, VSAM, and Adabas Gateways

   See: *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64*

5. Set up the Adabas data source

See: *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64*

6. Set up the data source metadata

   See: *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64*

7. Install Oracle Database Gateway for Adabas

   See: *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64*

8. Configure Oracle Database Gateway for Adabas

   See: *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64*

2

Gateway Features and Restrictions

After the gateway is installed and configured, you can use the gateway to access Adabas data, pass Adabas commands from applications to the Adabas database, perform distributed queries, and copy data.

This chapter contains the following sections:

- Accessing Adabas Data
- SQL Support
- Pass-Through Feature
- Handling Non-Relational Data
- Restrictions

Accessing Adabas Data

Oracle Connect for IMS, VSAM, and Adabas Gateways supports the following types of Adabas data sources:

- Adabas (Predict): The Adabas (Predict) data source uses Predict metadata.
- ADD-Adabas: The ADD-Adabas data source uses Oracle's internal repository (ADD), which is usually imported from Natural Data Definition Module (DDM) files. Alternatively, Predict metadata can be exported and subsequently imported into the ADD-Adabas data source.

For details on importing metadata, see the *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64* or the *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows*.

Both Adabas data sources provide very similar functions. The Adabas (Predict) data source supports one-phase transactions only. The ADD-Adabas data sources enjoys some added flexibility and functions resulting from the ability to customize the metadata in the ADD. Unless explicitly stated, all features and procedures described apply to both data sources.

Oracle supports data from Adabas version 6.2 and later on IBM z/OS platforms.

Exporting Predict Metadata into Adabas ADD

Some users who have Predict still prefer to use ADD to store metadata. The process of moving metadata from Predict to ADD is simple, although manual. It involves

exporting from Predict and importing to ADD. The process is carried out using the NAV_UTIL command-line interface.

In the following procedure, for the purposes of the example, note that the native qualifier is required. On the export side, the procedure generates all table definitions from a Predict data source called `adapredict` to an XML file. On the import side, the exported metadata is imported to a data source call `adaadd`.

Perform the following procedure to export Predict metadata into Adabas ADD.

1. After executing the NAVCMD Rexx script in USERLIB, execute the following NAV_UTIL command to export:

   ```
   Local> export table -native adapredict * 'ORACLE.XML.ADAPRED'
   ```

2. After executing the NAVCMD Ress script in USERLIB, execute the following NAV_UTIL command to import:

   ```
   Local> import adaadd 'ORACLE.XML.ADAPRED'
   ```

SQL Support

Oracle Database Gateway for Adabas supports SQL-based data access, allowing developers and applications to use this common and standard syntax for retrieving and updating Adabas data, and for using SELECT, INSERT, UPDATE, and DELETE statements.

While SQL is supported by relational database management systems, it is not available for legacy, non-relational data such as Adabas data. This means that Oracle Database Gateway for Adabas goes beyond the functionality provided by other Oracle gateways that connect to relational databases, as those merely deal with the translation and delegation of SQL to the target database. Oracle Database Gateway for Adabas takes care of the translation of the SQL statements into system and file specific access primitives, and their execution in an optimized manner, effectively serving as the SQL processor for the non-relational system.

In terms of SQL support, the gateway supports a wide range of SQL capabilities, from basic SQL statements to statements that include advanced features, such as joining of data from multiple Adabas sources, use of subselects, and support of data manipulation functions. In addition, the gateway optimizes query performance by supporting such advanced capabilities in where clauses, processing most of them before returning the data to Oracle.

Because Adabas is a non-relational system, its data model is not normalized. The gateway provides a complete normalization process that imports existing legacy metadata and produces a relational format that can be used by Oracle users. A key consideration in the normalization process takes care of the hierarchical data structures that are common in Adabas, such as arrays.

Going through the import process, the gateway translates the hierarchical structures and embedded arrays into a relational model that maps the hierarchy to several tables. In addition, the process automatically generates the necessary foreign constraints that can later be retrieved using regular Oracle Data Dictionary queries.

In respect to the data dictionary, the gateway provides all the necessary information although the actual data dictionary is not located in the Oracle database. The gateway supports standard data dictionary queries, making the interaction with the non-Oracle data completely transparent to the user, and taking care of translating the non-Oracle model into the Oracle standard data dictionary format including joins across non-Oracle data dictionary tables.

While the typical use case only supports the retrieval of information using SELECT statements, the gateway also supports data manipulation using INSERT, UPDATE, and DELETE statements. Furthermore, it supports advanced options such as distributed transactions.

To summarize, Oracle Database Gateway for Adabas offers robust SQL support and relational access to non-relational, proprietary, legacy Adabas data. The following topics provide more information about specific SQL features and their support by the gateway.

This section includes the following topics:

- Pass-Through Feature
- Naming Rules
- Data Types
- SQL Execution
- Locking

> **See Also:** Supported SQL Syntax and Functions on page B-1 for details; SQL Restrictions on page 2-6 for restrictions.

Pass-Through Feature

Adabas commands cannot be issued using the pass-through feature.

Naming Rules

Naming rule issues include the following:

- Rules for Naming Objects
- Case Sensitivity

Rules for Naming Objects

The concept of owner does not exist for objects from Adabas data sources. The userid of the dblink is not used to qualify the object. The owner field in the data dictionary tables (see Appendix C, "Data Dictionary" for details) is hardcoded as `ADABAS`. You must not use an explicit owner qualifier to reference Adabas tables. Using an explicit owner name results in a message like the following:

```
ORA-00942: table or view does not exist
```

> **See Also:** *Oracle Database Reference* and Adabas documentation for more information on naming objects and *Oracle Database Error Messages* for more information on error messages.

Case Sensitivity

Object names are not case sensitive. Both Oracle Studio for IMS, VSAM, and Adabas Gateways and the gateway automatically use upper case for data source metadata.

Data Types

Data type issues include the following:

- Binary Literal Notation
- Data Type Conversion

Binary Literal Notation

Oracle SQL uses hexadecimal digits surrounded by single quotes to express literal values being compared or inserted into columns defined as data type RAW.

This notation is not converted to syntax compatible with the Adabas VARBINARY and BINARY data types (a 0x followed by hexadecimal digits, surrounded by single quotes).

For example, the following statement is not supported:

```
SQL> INSERT INTO BINARY_TAB@Adabas VALUES ('0xff')
```

Where `BINARY_TAB` contains a column of data type VARBINARY or BINARY. Use bind variables when inserting into or updating VARBINARY and BINARY data types.

Data Type Conversion

Adabas does not support implicit date conversions. Such conversions must be explicit.

For example, the gateway issues an error for the following `SELECT` statement:

```
SELECT DATE_COL FROM TEST@Adabas WHERE DATE_COL = "1-JAN-2001";
```

To avoid problems with implicit conversions, add explicit conversions, as in the following:

```
SELECT DATE_COL FROM TEST@Adabas WHERE DATE_COL = TO_DATE("1-JAN-2001")
```

> **See Also:** Appendix A, "Data Type Conversion" for more information about restrictions on data types and *Oracle Database Error Messages* for more information on error messages.

SQL Execution

Query issues include the following:

- Row Selection
- Empty Strings
- Empty Bind Variables

Row Selection

Adabas evaluates a query condition for all selected rows before returning any of the rows. If there is an error in the evaluation process for one or more rows, no rows are returned even though the remaining rows satisfy the condition.

Oracle evaluates the query condition row-by-row and returns a row when the evaluation is successful. Rows are returned until a row fails the evaluation.

> **See Also:** *Oracle Database Error Messages* for more information on error messages.

Empty Strings

Oracle processes an empty string in a SQL statement as a null value. Adabas processes an empty string as an empty string.

Comparing to an empty string

The Gateway passes literal empty strings to the Adabas database without any conversion. If you intended an empty string to represent a null value, Adabas does not process the statement that way; it uses the empty string.

You can avoid this problem by using NULL or IS NULL in the SQL statement instead of the empty string syntax, as in the following example:

```
SELECT * from "emp"@Adabas where "ename" IS NULL;
```

Selecting an empty string

For VARCHAR columns, the gateway returns an empty string to the Oracle Database as NULL value.

For CHAR columns, the gateway returns the full size of the column with each character as empty space (' ').

Empty Bind Variables

For VARCHAR bind variables, the gateway passes empty bind variables to the Adabas database as a NULL value.

Locking

The locking model for an Adabas database differs significantly from the Oracle model. The gateway depends on the underlying Adabas behavior, so Oracle applications that access Adabas through the gateway can be affected by the following possible scenarios:

- Read access may block write access.
- Write access may block read access.
- Statement-level read consistency is not guaranteed.

> **See Also:** Adabas documentation for information about the Adabas locking model.

Handling Non-Relational Data

Non-relational data sources (excluding Adabas when Predict metadata is used) require metadata, which is kept separately from the data itself. This metadata is stored as a data source definition in a data source repository, on the machine where the data source is defined. It lets you access the data from a non-relational database with SQL commands.

When the non-relational data contains arrays, these arrays can be exposed as follows:

Once the metadata is imported, the data from the non-relational data source can be normalized to maintain transparency.

- As virtual views. This method generates a virtual view for every array in the parent record that contains all the array members.
- As a single table. This method maps all the record fields of the non-relational file to a single table that contains both parent and child records.

Restrictions

The following sections describe the restrictions and include suggestions for dealing with them if possible:

- SQL Restrictions
- Adabas Restrictions

SQL Restrictions

Restrictions related to SQL are described in the following sections:

- Unsupported SQL Functions
- SQL Syntax
- Transaction Capability
- COMMIT or ROLLBACK in PL/SQL Cursor Loops
- Pass-Through Feature
- Database Links
- Data Types
- Encrypted Format Login
- Data Dictionary Views and PL/SQL

Unsupported SQL Functions

When an unsupported SQL function is used in an UPDATE, DELETE, or INSERT statement, an error occurs.

See Supported SQL Syntax and Functions on page B-1 for a list of the supported functions.

SQL Syntax

Table 2–1 lists the restrictions that apply to SQL syntax.

Table 2–1 SQL Syntax Restrictions

Syntax	Restriction
WHERE CURRENT OF clause	`UPDATE` and `DELETE` statements with the `WHERE CURRENT OF` clause are not supported by the gateway because they rely on the Oracle ROWID implementation. To update or delete a specific row through the gateway, a condition style `WHERE` clause must be used.
CONNECT BY clause	The gateway does not support the `CONNECT BY` clause in a `SELECT` statement.
ROWID	The Oracle ROWID implementation is not supported.
Subqueries in INSERT statement	Subqueries of `INSERT` statements cannot use multiple aliases for the same table. For example, the following statement is not supported: `SQL> INSERT INTO "emp_target"@Adabas SELECT a."empno" FROM "emp_source"@Adabas a, "emp_source"@Adabas b WHERE b."empno"=9999`
EXPLAIN PLAN statement	The `EXPLAIN PLAN` statement is not supported.

Table 2–1 (Cont.) SQL Syntax Restrictions

Syntax	Restriction
Date arithmetic	The following SQL expressions do not function correctly with the gateway: *date + number* *number + date* *date - number* *date1 - date2* Statements with the preceding expressions are sent to the Adabas database without any translation. Since Adabas does not support these date arithmetic functions, the statements return an error. **See Also:** *Oracle Database Error Messages* for more information on error messages.
String functions	If you concatenate numeric literals using the "||" operator when using the gateway to query a Adabas database, the result is an arithmetic addition. For example, the result of the following statement is 18: `SQL> SELECT 9 || 9 FROM DUAL@Adabas;` The result is 99 when using Oracle to query an Oracle database.
Schema names and PL/SQL	If you do not prefix a Adabas database object with its schema name in a SQL statement within a PL/SQL block, the following error message occurs: `ORA-6550 PLS-201 Identifier` *table_name* `must be declared.` Change the SQL statement to include the schema name of the object. **See Also:** *Oracle Database Error Messages* for more information on error messages.

Transaction Capability

The gateway does not support savepoints. If a distributed update transaction is under way involving the gateway and a user attempts to create a savepoint, the following error occurs:

`ORA-02070: database` *dblink* `does not support savepoint in this context`

The Adabas data source supports only local transactions.

> **See Also:** *Oracle Database Error Messages* for more information on error messages.

COMMIT or ROLLBACK in PL/SQL Cursor Loops

Any COMMIT or ROLLBACK issued in a PL/SQL cursor loop closes all open cursors, which can result in the following error:

```
ORA-1002:  fetch out of sequence
```

To prevent this error, move the COMMIT or ROLLBACK statement outside the cursor loop.

> **See Also:** *Oracle Database Error Messages* for more information on error messages.

Pass-Through Feature

Adabas commands cannot be issued using the pass-through feature.

Database Links

The gateway is not multithreaded and cannot support shared database links. Each gateway session spawns a separate gateway process and connections cannot be shared.

Data Types

The FLOAT data type precision is (7).

Encrypted Format Login

Oracle Database no longer supports the initialization parameter `DBLINK_ENCRYPT_LOGIN`. In version 7.3 and earlier, this parameter's default `true` value prevented the password for the login user ID from being sent over the network (in the clear). Later versions automatically encrypt the password.

Data Dictionary Views and PL/SQL

You cannot refer to data dictionary views in SQL statements that are inside a PL/SQL block.

Adabas Restrictions

When accessing Adabas data, the following restrictions apply:

- UPDATE operations on arrays are not supported. See Normalizing Non-Relational Data on page 3-1 for details on handling arrays.
- When you perform an INSERT operation with an incomplete list of values, the columns that are described as NOT NULL take on the default value of the specific data type. The following table lists the default value for each data type.

Data Type	Default Value
numeric	0
char	spaces
date	1-1-0000

3

Normalizing Non-Relational Data

This section describes the support methods that Oracle Connect for IMS, VSAM, and Adabas Gateways applies to normalize non-relational data. It includes the following topics:

- Overview of Normalizing Non-Relational Data
- Representing Metadata
- Methods of Handling Arrays

> **See Also:** *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64* for information on importing metadata into Oracle Studio for IMS, VSAM, and Adabas Gateways.

Overview of Normalizing Non-Relational Data

Oracle Connect for IMS, VSAM, and Adabas Gateways exposes a purely relational front end through the HOA API. However, it connects to non-relational data sources, which include non-relational data models. As such, Oracle Connect for IMS, VSAM, and Adabas Gateways provides a logical mapping that exposes the non-relational constructs in a relational manner. The most prevalent problem in this domain is the issue of arrays, which is described in this section.

About Arrays

An array is a group of similar elements of the same size. Arrays contain a series of data elements that are of the same data type, which can be simple or complex (group). A specific element is defined and accessed by its position in the array, which is provided by an index.

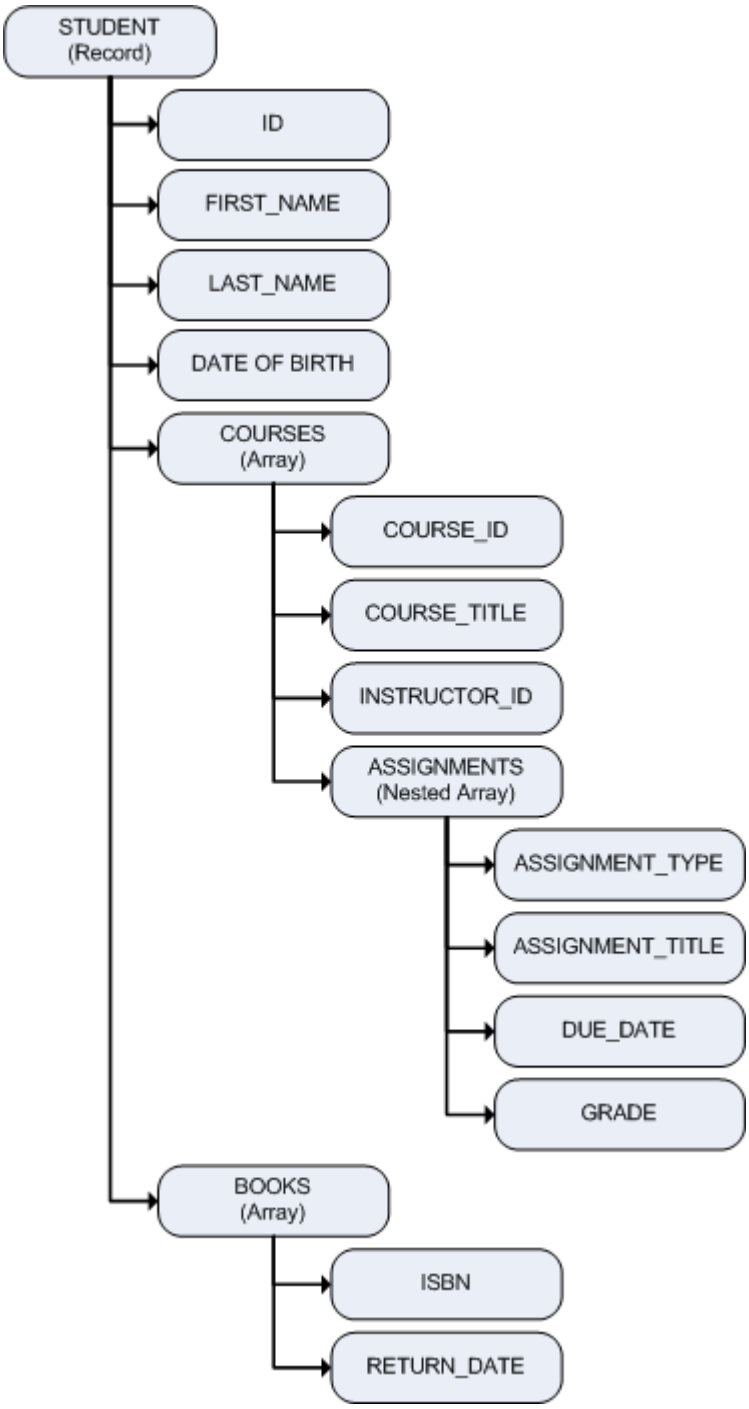
Arrays are a convenient way to store a fixed amount of data that is accessed in an unpredictable fashion. They are not efficient, however, when you need to insert or delete individual elements of the array.

Representing Metadata

Before looking at the different methods of handling arrays, you should understand how metadata is represented in Oracle Studio for IMS, VSAM, and Adabas Gateways.

Figure 3–1 shows an example record with arrays and nested arrays.

Figure 3–1 Arrays and Nested Arrays

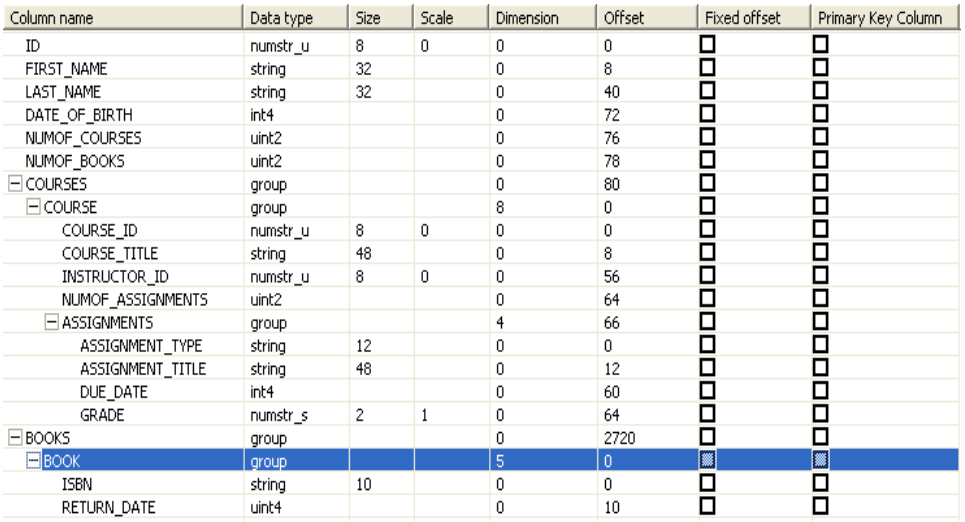


When you import this metadata into Oracle Studio for IMS, VSAM, and Adabas Gateways, the import process creates a data dictionary definition that is equivalent to the original structure, usually mapping the fields one to one. The import process also writes all primary and foreign key definitions to the ALL_CONSTRAINTS table.

Note: When you work with the Adabas (Predict) data source, you can only generate the metadata for the arrays by editing the specific table.

Oracle Studio for IMS, VSAM, and Adabas Gateways represents the flattened view of the metadata on the table editor's Columns tab, as shown in Figure 3–2.

Figure 3–2 Representation of Metadata on the Columns tab in Oracle Studio for IMS, VSAM, and Adabas Gateways



Column name	Data type	Size	Scale	Dimension	Offset	Fixed offset	Primary Key Column
ID	numstr_u	8	0	0	0	☐	☐
FIRST_NAME	string	32		0	8	☐	☐
LAST_NAME	string	32		0	40	☐	☐
DATE_OF_BIRTH	int4			0	72	☐	☐
NUMOF_COURSES	uint2			0	76	☐	☐
NUMOF_BOOKS	uint2			0	78	☐	☐
⊟ COURSES	group			0	80	☐	☐
⊟ COURSE	group			8	0	☐	☐
COURSE_ID	numstr_u	8	0	0	0	☐	☐
COURSE_TITLE	string	48		0	8	☐	☐
INSTRUCTOR_ID	numstr_u	8	0	0	56	☐	☐
NUMOF_ASSIGNMENTS	uint2			0	64	☐	☐
⊟ ASSIGNMENTS	group			4	66	☐	☐
ASSIGNMENT_TYPE	string	12		0	0	☐	☐
ASSIGNMENT_TITLE	string	48		0	12	☐	☐
DUE_DATE	int4			0	60	☐	☐
GRADE	numstr_s	2	1	0	64	☐	☐
⊟ BOOKS	group			0	2720	☐	☐
⊟ BOOK	group			5	0	▩	▩
ISBN	string	10		0	0	☐	☐
RETURN_DATE	uint4			0	10	☐	☐

Table 3–1 describes the different columns shown in Figure 3–2.

The (+) to the left of a column indicates a group field. This type of field usually has a Dimension value. This value is not mandatory, but it optimizes the access to an array. You can click (+) to display the group members. In Figure 3–2, all groups are expanded.

Table 3–1 Metadata Column Tab Definition

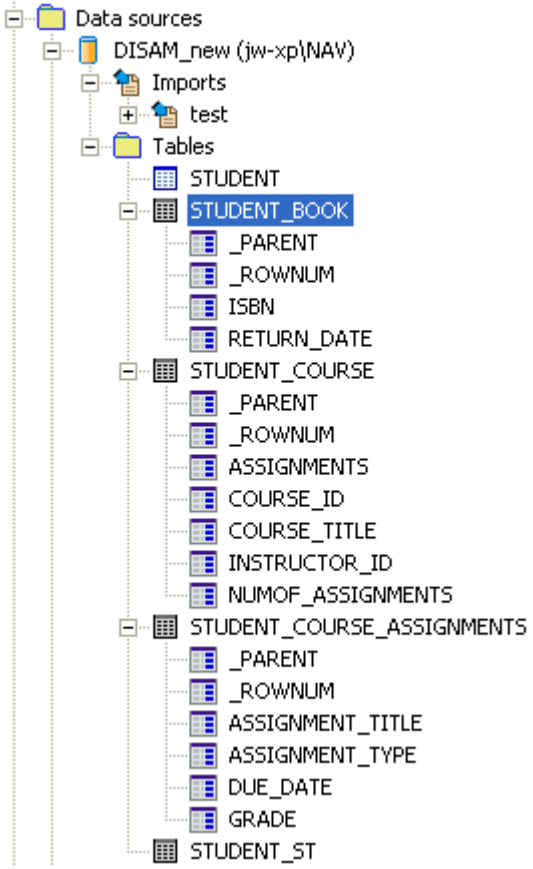
Column	Description
Column name	The name of the column.
Data type	The data type of the column. Selecting this field displays a drop-down box listing the possible data types.
Size	The size of the column for data types of a non-fixed size.
Scale	The information entered in this field depends on the data type: For decimal data types, this is the number of digits to the right of the decimal place. This number must not be greater than the number of digits. The default value is 0. For scaled data types, this is the total number of digits. The number must be negative.
Dimension	The maximum number of entries of an array. An array has a dimension other than zero. This value is not mandatory, but it optimizes the access to an array.
Offset	Not relevant for arrays.
Fixed offset	Not relevant for arrays.

Table 3–1 (Cont.) Metadata Column Tab Definition

Column	Description
Primary Key Column	The column is part of the table's primary key.

The tree in the Metadata view displays a normalized view of the tables, as shown in Figure 3–3, where STUDENT_BOOK, STUDENT_COURSE, and STUDENT_COURSE_ASSIGNMENTS are virtual views and STUDENT_ST is a single table, also called sequential view.

Figure 3–3 Normalized View of the Tables in the Metadata View



The following SQL query on the Oracle data dictionary produces the result shown in Example 3–1. In this query, *data source* is the name of the data source from which the metadata was created. The result shows the metadata of the original table after virtual arrays were created.

```
SQL> select TABLE_NAME,COLUMN_NAME,DATA_TYPE from ALL_TAB_COLUMNS@dg4[data_source]
where table_name like 'STUDENT%';
```

Example 3–1 SQL Query Result

```
TABLE_NAME                     COLUMN_NAME                DATA_TYPE
------------------------------ -------------------------- --------------------
STUDENT                        ID                         NUMBER
STUDENT                        FIRST_NAME                 CHAR
STUDENT                        LAST_NAME                  CHAR
STUDENT                        DATE_OF_BIRTH              CHAR
```

```
STUDENT                        NUMOF_COURSES              NUMBER
STUDENT                        NUMOF_BOOKS                NUMBER
STUDENT_BOOK                   BOOK_ROWNUM                NUMBER
STUDENT_BOOK                   ISBN                       CHAR
STUDENT_BOOK                   RETURN_DATE                CHAR
STUDENT_COURSE                 COURSE_ROWNUM              NUMBER
STUDENT_COURSE                 COURSE_ID                  NUMBER
STUDENT_COURSE                 COURSE_TITLE               CHAR
STUDENT_COURSE                 INSTRUCTOR_ID              NUMBER
STUDENT_COURSE                 NUMOF_ASSIGNMENTS          NUMBER
STUDENT_COURSE_ASSIGNMENTS     COURSE_ROWNUM              NUMBER
STUDENT_COURSE_ASSIGNMENTS     ASSIGNMENTS_ROWNUM         NUMBER
STUDENT_COURSE_ASSIGNMENTS     ASSIGNMENT_TYPE            CHAR
STUDENT_COURSE_ASSIGNMENTS     ASSIGNMENT_TITLE           CHAR
STUDENT_COURSE_ASSIGNMENTS     DUE_DATE                   CHAR
STUDENT_COURSE_ASSIGNMENTS     GRADE                      NUMBER
STUDENT_ST                     __LEVEL                    VARCHAR2
STUDENT_ST                     __SEQUENCE                 NUMBER
STUDENT_ST                     COURSE_ROWNUM              NUMBER
STUDENT_ST                     ASSIGNMENTS_ROWNUM         NUMBER
STUDENT_ST                     BOOK_ROWNUM                NUMBER
STUDENT_ST                     ID                         NUMBER
STUDENT_ST                     FIRST_NAME                 CHAR
STUDENT_ST                     LAST_NAME                  CHAR
STUDENT_ST                     DATE_OF_BIRTH              CHAR
STUDENT_ST                     NUMOF_COURSES              NUMBER
STUDENT_ST                     NUMOF_BOOKS                NUMBER
STUDENT_ST                     COURSE_ID                  NUMBER
STUDENT_ST                     COURSE_TITLE               CHAR
STUDENT_ST                     INSTRUCTOR_ID              NUMBER
STUDENT_ST                     NUMOF_ASSIGNMENTS          NUMBER
STUDENT_ST                     ASSIGNMENT_TYPE            CHAR
STUDENT_ST                     ASSIGNMENT_TITLE           CHAR
STUDENT_ST                     DUE_DATE                   CHAR
STUDENT_ST                     GRADE                      NUMBER
STUDENT_ST                     ISBN                       CHAR
STUDENT_ST                     RETURN_DATE                CHAR
41 rows selected.
```

Methods of Handling Arrays

Oracle Studio for IMS, VSAM, and Adabas Gateways lets you handle arrays by using the following methods:

- Virtual Views
- Sequential Flattening (Bulk Load of Array Data)

> **Note:** Objects from virtual views and sequential flattening are described by Oracle Gateways as tables.

See Chapter 4, "Setting the Array Handling Policy" for information on how to define array handling settings.

Virtual Views

Exposing arrays as virtual views is a commonly used technique to handle arrays. It generates a virtual view for every array in the parent record that contains all the array

members. Virtual views contain primary key fields from the parent to connect the parent and the virtual view. Optionally, they can also contain all fields from the parent table.

During the import process or when you set the virtual array policy on the table level, Oracle Studio for IMS, VSAM, and Adabas Gateways generates virtual views and names them by appending the array name to the parent name. When an array includes another array, the name of the resulting virtual table consists of the parent name, the array name, and the name of the nested array, as follows:

```
parentName_arrayName_nestedArrayName
```

For example, a parent table called `STUDENT` with an array called `COURSE` and a nested array called `ASSIGNMENTS` is represented by the virtual view `STUDENT_COURSE_ASSIGNMENTS` (see Figure 3–5).

The number of nested-array levels is not limited.

Virtual views include the following:

- The array member columns from the original structure.
- The fields from the parent's first unique key, or all parent fields, depending on the selection you make during the import process or when setting the virtual array policy on the table level.

  If all parent fields are included in the virtual view, the parent's indexes are available in the view definition and can be used for efficient optimization strategies.

  Note: Inherited keys lose their uniqueness in the virtual view.

- If the view does not include all parent fields, the primary key fields (if the primary key is not the parent's first unique key).
- If selected, a column called `<array_name>_ROWNUM`, which identifies the row in the array.

The unique key and `<array_name>_ROWNUM` columns are generated automatically. Together, they uniquely identify each row in the virtual view and form a unique key.

Figure 3–4 shows the `STUDENT_BOOKS` virtual view with the two array member columns (`ISBN` and `RETURN_DATE`) and the column that identifies the row in the array (`BOOK_ROWNUM`).

Figure 3–4 Virtual View STUDENT_BOOKS

BOOK_ROWNUM	ISBN	RETURN_DATE
1	1234	2004-05-02
1	5678	2005-03-11
1	5334	2003-12-29
2	7843	2003-12-29
1	8039	2003-12-15
2	4930	2003-12-12
1	4738	2003-04-23
2	2849	2003-06-18
1	2849	2003-09-15
2	8943	2004-01-13
3	7832	2004-01-20
1	1820	2003-10-30
2	9493	2003-11-16
3	2819	2003-11-16

Oracle Studio for IMS, VSAM, and Adabas Gateways also maintains primary and foreign key definitions that connect between the parent table and the array tables. This allows graphical tools to easily match parent and array.

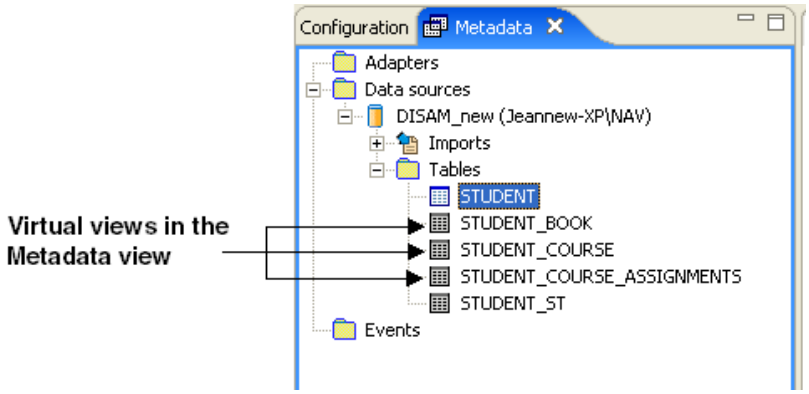
When working with virtual views, consider the following limitations:

- Virtual views are read-only.
- Virtual views do not support arrays within variants that have a selector field.

Including all parent fields in the virtual view greatly reduces the need for performing join operations because this in itself is an implicit join. In general, the query processor can devise efficient access strategies because Oracle Connect for IMS, VSAM, and Adabas Gateways copies all relevant indexes from the parent to the virtual view.

Oracle Studio for IMS, VSAM, and Adabas Gateways indicates virtual views by using a different colored icon in the Metadata view, as shown in Figure 3–5.

Figure 3–5 Display of Virtual Views in Oracle Studio for IMS, VSAM, and Adabas Gateways



Sequential Flattening (Bulk Load of Array Data)

Performing a bulk load of complex data from a non-relational system to a relational database requires a carefully thought-out algorithm that keeps I/O operations at a minimum.

In a bulk load scenario, a method such as Virtual Views requires a full scan of the physical file for every single array. An efficient method of performing this task presents a kind of row-wise normalization, called sequential flattening. This method reads all data in the physical file in a single scan.

Sequential flattening replaces arrays in a non-relational system by a sequence of rows. It maps all the record fields of the non-relational file to a single table that contains both parent and child records. In this way, sequential flattening enables the reception of a stream of data by using a single `SELECT` statement.

The sequentially flattened view of a complex table is referred to as a *single table* or *sequential view*. You can choose to create a sequential view in Oracle Studio for IMS, VSAM, and Adabas Gateways by selecting the **Generate sequential view** check box during the Metadata Model Selection step of the Metadata Import procedure. The single table is read-only.

> **See Also:** *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64* for details on importing metadata into Oracle Studio for IMS, VSAM, and Adabas Gateways.

The flattened table is called `<table>_ST`, where `<table>` is the name of the parent table and `ST` indicates a single table. For example, if a parent table is called `STUDENT`, the single table, or sequential view, is called `STUDENT_ST`.

The structure of the single table is identical to the original table's structure, except that Oracle Connect for IMS, VSAM, and Adabas Gateways removes all array dimensions and adds some control fields. When reading a record, Oracle Connect for IMS, VSAM, and Adabas Gateways performs a tree traversal of the parent and its array hierarchy. Each record in the resulting recordset deals with a specific array member; other arrays are nulled out.

The sequentially flattened single table includes the following columns:

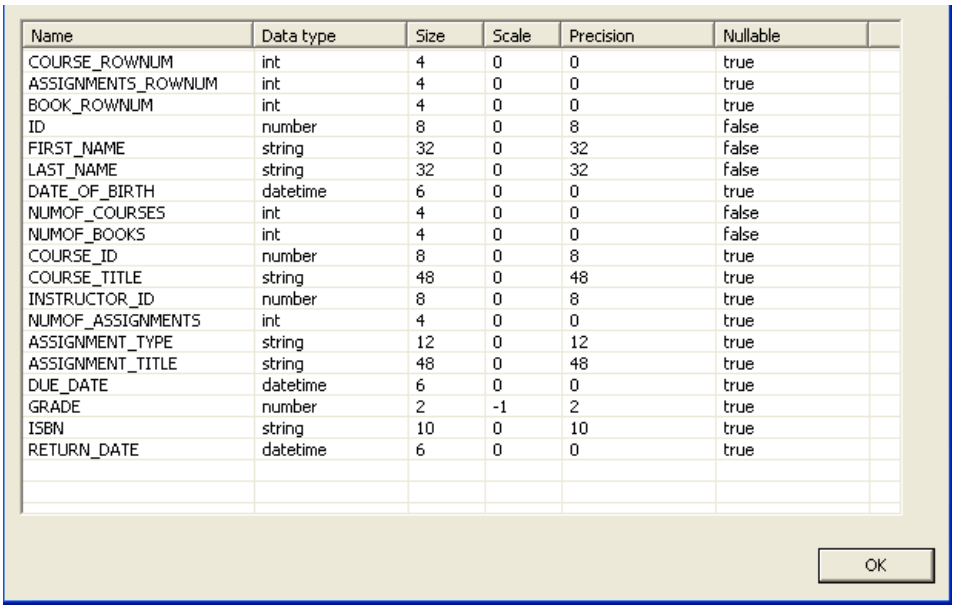
- The parent fields, that is the non-repeating fields.
- The array fields for all arrays within the parent.
- For each array, an optional column called `<array_name>_ROWNUM`, which identifies the row in the array. This column is generated automatically for the array.

The sequentially flattened single table includes a record (row) for each array record.

See Figure 3–1 for an illustration of arrays and nested arrays.

Figure 3–6 shows the metadata that sequential flattening produces for a data source with three arrays (`COURSE`, `ASSIGNMENTS`, and `BOOK`) in the SQL View window. The window presents read-only information about each of the columns in the table.

Figure 3–6 SQL View of the Single Table's Metadata



Name	Data type	Size	Scale	Precision	Nullable
COURSE_ROWNUM	int	4	0	0	true
ASSIGNMENTS_ROWNUM	int	4	0	0	true
BOOK_ROWNUM	int	4	0	0	true
ID	number	8	0	8	false
FIRST_NAME	string	32	0	32	false
LAST_NAME	string	32	0	32	false
DATE_OF_BIRTH	datetime	6	0	0	true
NUMOF_COURSES	int	4	0	0	false
NUMOF_BOOKS	int	4	0	0	false
COURSE_ID	number	8	0	8	true
COURSE_TITLE	string	48	0	48	true
INSTRUCTOR_ID	number	8	0	8	true
NUMOF_ASSIGNMENTS	int	4	0	0	true
ASSIGNMENT_TYPE	string	12	0	12	true
ASSIGNMENT_TITLE	string	48	0	48	true
DUE_DATE	datetime	6	0	0	true
GRADE	number	2	-1	2	true
ISBN	string	10	0	10	true
RETURN_DATE	datetime	6	0	0	true

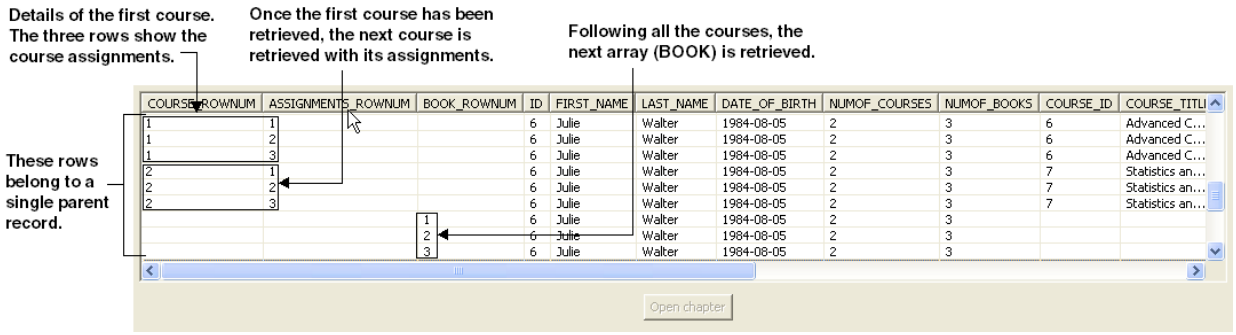
The following table describes the information presented in this window.

Table 3–2 SQL View Window

Information Type	Description
Name	The name of the column.
Data Type	The data type supported by that column. For example, string or integer.
Size	Indicates the maximum size allowable for the data in the column. The size is in standard units for the data type. For example, a string with size 40 can have no more than forty characters.
Scale	Indicates the number of digits allowed after the decimal point for a numeric value.
Precision	Indicates the total number of digits allowed for a numeric value in the column. If the value has a scale of one or more, then the total number of digits allowed before the decimal point is the precision value minus the scale value. For example, a value with precision 4 and scale 2 can be no larger than 99.99.
Nullable	Indicates whether the column can have a null value. If `True`, the column is nullable.

Figure 3–7 shows the actual single table after running a SELECT clause. It contains a column for each row in the preceding SQL view.

Figure 3–7 STUDENT_ST with All Parent and Child Records



An SQL describe of this single table produces the following result:

```
Name                                      Null?            Type
----------------------------------------- ---------------- ------------------
__LEVEL                                   NOT NULL         VARCHAR2(64)
__SEQUENCE                                NOT NULL         NUMBER(11)
COURSE_ROWNUM                                              NUMBER(10)
ASSIGNMENTS_ROWNUM                                         NUMBER(10)
BOOK_ROWNUM                                                NUMBER(10)
ID                                        NOT NULL         NUMBER(8)
FIRST_NAME                                NOT NULL         CHAR(32)
LAST_NAME                                 NOT NULL         CHAR(32)
DATE_OF_BIRTH                             NOT NULL         CHAR(8)
NUMOF_COURSES                             NOT NULL         NUMBER(10)
NUMOF_BOOKS                               NOT NULL         NUMBER(10)
COURSE_ID                                                  NUMBER(8)
COURSE_TITLE                                               CHAR(48)
INSTRUCTOR_ID                                              NUMBER(8)
NUMOF_ASSIGNMENTS                                          NUMBER(10)
ASSIGNMENT_TYPE                                            CHAR(12)
ASSIGNMENT_TITLE                                           CHAR(48)
DUE_DATE                                                   CHAR(8)
GRADE                                                      NUMBER(2,1)
ISBN                                                       CHAR(10)
RETURN_DATE                                                CHAR(8)
```

This table shows all the columns of the single table.

4

Setting the Array Handling Policy

This chapter contains the following sections:

- Setting the Default Behavior for Array Handling
- Modifying the Array Handling Policy for a Specific Table

Setting the Default Behavior for Array Handling

You can define the default behavior for array handling at the data source level when configuring the data source properties.

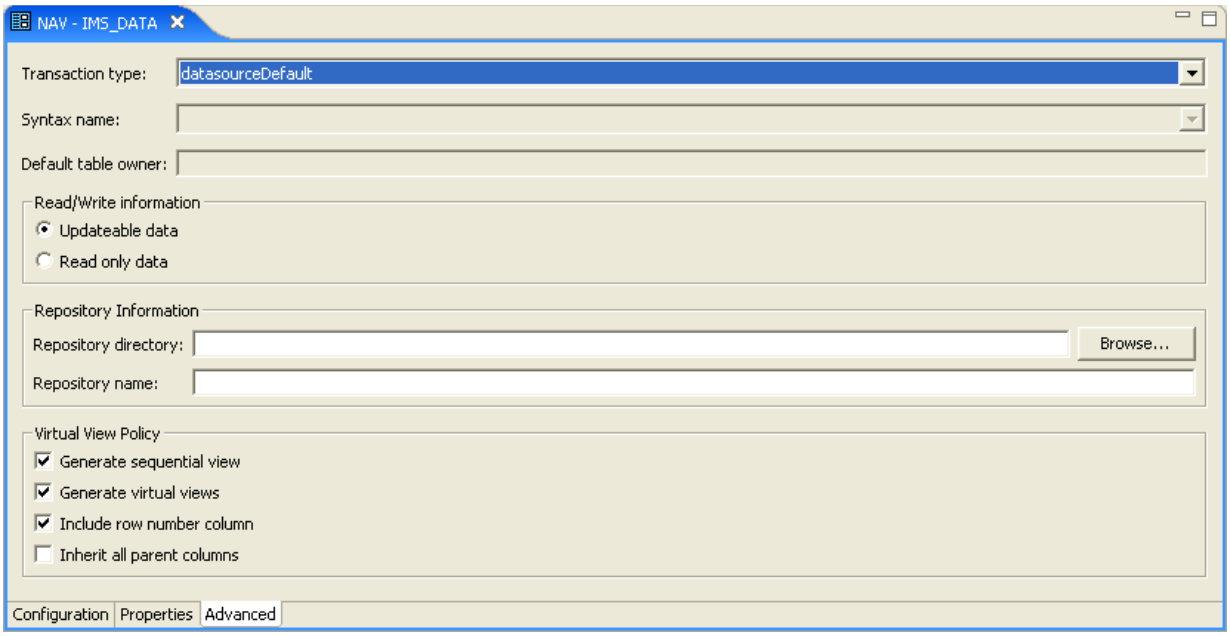
See *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64* for details on configuring data source properties.

Perform the following steps in Oracle Studio for IMS, VSAM, and Adabas Gateways to set the default array handling behavior:

1. In the Design Perspective Configuration Viewexpand the **Machine** folder and then expand the machine with the data source where you want to set the array handling behavior.
2. Expand the binding with the data source you are working with.
3. Expand the **Data Sources** folder and then right-click the data source and select **Edit Data Source**.

   The Configuration Properties screen opens.
4. On the **Advanced** tab, in the **Virtual View Policy** section, configure how arrays will be handled by selecting the relevant check boxes. The following options are available:
   - **Generate sequential view**: Select this option if you want to map a non-relation file to a single table.
   - **Generate virtual views**: Select this option if you want to have an individual table created for every array in the non-relational file.
   - **Include row number column**: Select this option if you want to include a column that specifies the row number in the virtual or sequential view.
   - **Inherit all parent columns**: Select this option if you want the virtual views to include all the columns of the parent record.

Figure 4–1 Setting the Default Behavior for Handling Arrays



5. Save your settings.

Modifying the Array Handling Policy for a Specific Table

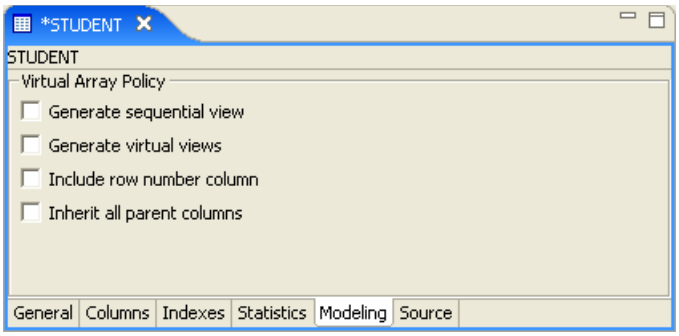
You can also change the array handling policy on the table level. Perform the following steps to modify the policy for a specific table:

1. In the Design Perspective Configuration Viewexpand the **Machine** folder and then expand the machine with the data source where you want to set the array handling behavior.
2. Expand the binding with the data source you are working with.
3. Expand the data sources folder, then right-click the data source and select **Edit metadata**.

   The Metadata view opens on the node of the respective data source.
4. Expand the data source node, and then expand the **Tables** node.
5. Double-click the table whose array handling settings you want to modify.

   The table editor opens.
6. On the **Modeling** tab, configure how arrays will be handled by selecting the relevant check boxes. The following options are available:
   - **Generate sequential view**: Select this option if you want to map a non-relation file to a single table.
   - **Generate virtual views**: Select this option if you want to have an individual table created for every array in the non-relational file.
   - **Include row number column**: Select this option if you want to include a column that specifies the row number in the virtual or sequential view.
   - **Inherit all parent columns**: Select this option if you want the virtual views to include all the columns of the parent record.

Figure 4–2 Modifying the Array Handling Policy on the Table Level



7. Save your settings.

Oracle Studio for IMS, VSAM, and Adabas Gateways creates new virtual views.

A

Data Type Conversion

This appendix contains the following section:

- Conversion of Predict Metadata
- Conversion of Metadata from DDM Files

Conversion of Predict Metadata

The gateway converts Predict metadata data types in the advanced data dictionary (ADD) to Oracle data types as described in Table A–1.

Table A–1 Predict Metadata

ADD Data Type	Oracle Data Type	Details
A	VARCHAR2	A null-terminated string of alphanumeric characters; maximum length must be specified. An extra byte is required for the null flag.
B1	Number(3)	Unsigned byte integer. Size: 1 byte Range: 0 to +254
B2	Number(5)	Unsigned word integer. Size: 2 bytes Range: 0 to +65535
B4	Number(10)	Unsigned long integer. Size: 4 bytes Range: 0 to +4,294,967,294
F1	Number(3)	Signed byte integer. Size: 1 byte Signed range: -128 to +127
F2	Number(5)	Signed word integer. Size: 2 bytes Signed Range: -32768 to +32767
F4	Number(10)	Signed long integer. Size: 4 bytes Range: -2147483648 to 2147483647

Table A–1 (Cont.) Predict Metadata

ADD Data Type	Oracle Data Type	Details
F8	Number(19)	Signed long integer. Size: 8 bytes Range: -9223372036854775808 to +9223372036854775807
G4	FLOAT(23)	Single floating-point number (F_FLOAT). Size: 4 bytes Range: 0.29E-38 to 1.7E38 Precision: 6 digits
G8	FLOAT(49)	Double floating-point number (D_FLOAT) Size: 8 bytes Range: 029E-38 to 1.7E38 Precision: 16 digits
L	NUMBER(3)	Unsigned byte integer. Size: 1 byte Values: 1 for true, 0 for false
D	DATE	ODBC date format.
T	DATE	ODBC time format.
P	NUMBER	Packed decimal Maximum number of digits: 31 Maximum fractions: 11 Length = int (number of digits/2) + 1 byte
N, U	NUMBER	Signed numeric string. A right overpunched sign is implemented. Maximum number of digits: 31 Maximum fractions: 11

Conversion of Metadata from DDM Files

When you import metadata from a DDM file, the gateway maps the metadata data types as described in Table A–2.

Table A–2 Metadata from DDM Files

DDM/Natural Data Type	ADD-Adabas Data Type	Oracle Data Type	Details
A	string	VARCHAR2	A null-terminated string of alphanumeric characters; maximum length must be specified. An extra byte is required for the null flag.
B1	Uint1	Number(3)	Unsigned byte integer. Size: 1 byte Range: 0 to +254
B2	Uint2	Number(5)	Unsigned word integer. Size: 2 bytes Range: 0 to +65535

Table A–2 (Cont.) Metadata from DDM Files

DDM/Natural Data Type	ADD-Adabas Data Type	Oracle Data Type	Details
B4	Uint4	Number(10)	Unsigned long integer. Size: 4 bytes Range: 0 to +4,294,967,294
I1	Int1	Number(3)	Signed byte integer. Size: 1 byte Signed range: -128 to +127
I2	Int2	Number(5)	Signed word integer. Size: 2 bytes Signed range: -32768 to +32767
I4	Int4	Number(10)	Signed long integer. Size: 4 bytes Range: -2147483648 to 2147483647
I8	Int8	Number(19)	Signed long integer. Size: 8 bytes Range: -9223372036854775808 to +9223372036854775807
F4	single	FLOAT(23)	Single floating-point number (F_FLOAT). Size: 4 bytes Range: 0.29E-38 to 1.7E38 Precision: 6 digits
F8	dfloat	Number(5)	Double floating-point number (D_FLOAT) Size: 8 bytes Range: 0.29E-38 to 1.7E38 Precision: 16 digits
L	Unit1	Number(3)	Unsigned byte integer. Size: 1 byte Values: 1 for true, 0 for false
D	ada_d_time	DATE	ODBC date format.
T	ada_time	DATE	ODBC time format.
P	decimal	NUMBER	Packed decimal Maximum number of digits: 31 Maximum fraction: 11 Length = int (number of digits/2) + 1 byte
N, U	ada_numstr_s	NUMBER	Signed numeric string Maximum number of digits: 31 Maximum fractions: 11

B

Supported SQL Syntax and Functions

This appendix contains the following sections:

- Supported SQL Statements
- Oracle Functions

> **See Also:** *Oracle Database SQL Reference* for detailed descriptions of keywords, parameters, and options and the Adabas documentation for details of executing SQL statements in a transaction.

Supported SQL Statements

With a few exceptions, the gateway provides full support for Oracle `DELETE`, `INSERT`, `SELECT`, and `UPDATE` statements.

The gateway does not support Oracle data definition language (DDL) statements. No form of the Oracle `ALTER`, `CREATE`, `DROP`, `GRANT`, or `TRUNCATE` statements can be used. Instead, for `ALTER`, `CREATE`, `DROP`, and `GRANT` statements, use the pass-through feature of the gateway if you need to use DDL statements against the Adabas database.

Note: `TRUNCATE` cannot be used in a pass-through statement.

DELETE

The `DELETE` statement is fully supported. However, only Oracle functions supported by Adabas can be used.

> **See Also:** Functions Supported by Adabas for a list of supported functions.

INSERT

The `INSERT` statement is fully supported. However, only Oracle functions supported by Adabas can be used.

> **See Also:** Functions Supported by Adabas for a list of supported functions.

SELECT

The `SELECT` statement is fully supported, with these exceptions:

- CONNECT BY *condition*
- NOWAIT
- START WITH *condition*
- WHERE CURRENT OF

UPDATE

The `UPDATE` statement is fully supported. However, only Oracle functions supported by Adabas can be used. Also, you cannot have SQL statements in the subquery that refer to the same table name in the outer query. Subqueries are not supported in the SET clause.

> **See Also:** Functions Supported by Adabas for a list of supported functions.

Oracle Functions

All functions are evaluated by the Adabas database after the gateway has converted them to Adabas SQL.

Functions Not Supported by Adabas

Oracle SQL functions with no equivalent function in Adabas are not supported in `DELETE`, `INSERT`, or `UPDATE` statements, but are evaluated by the Oracle Database if the statement is a `SELECT` statement. That is, the Oracle Database performs post-processing of `SELECT` statements sent to the gateway.

If an unsupported function is used in a `DELETE`, `INSERT`, or `UPDATE`, statement, the following Oracle error occurs:

```
ORA-02070: database db_link_name does not support function in this context
```

Functions Supported by Adabas

The gateway translates the following Oracle Database functions in SQL statements to their equivalent Adabas functions:

- Arithmetic Operators
- Comparison Operators
- Group Functions
- String Functions
- Other Functions

Arithmetic Operators

The following table specifies the arithmetic operators.

Oracle	Adabas
+	+
-	-
*	*
/	/

Comparison Operators

The following table specifies the comparison operators.

Oracle	Adabas
=	=
>	>
<	<
>=	>=
<=	<=
<>, !=, ^=	<>, !=
IS NOT NULL	IS NOT NULL
IS NULL	IS NULL

Group Functions

The following table specifies the group functions.

Oracle	Adabas
AVG	AVG
COUNT	COUNT
MAX	MAX
MIN	MIN
SUM	SUM

String Functions

The following table specifies the string functions.

Oracle	Adabas
||	+ (*expression1* + *expression2*)
ASCII	ASCII
CHR	CHAR
INSTR	CHARINDEX
LENGTH	CHAR_LENGTH
LOWER	LOWER
SUBSTR (second argument cannot be a negative number)	SUBSTRING
UPPER	UPPER

Other Functions

The following table specifies other functions.

Oracle	Adabas
ABS	ABS
CEIL	CEILING
COS	COS
EXP	EXP
FLOOR	FLOOR
LN	LOG
LOG	LOG10
MOD	%
NVL	IS NULL
POWER	POWER
ROUND	ROUND
SIN	SIN
SQRT	SQRT
TAN	TAN

C

Data Dictionary

The Oracle Database Gateway for Adabas translates a query that refers to an Oracle Database data dictionary table into a query that retrieves the data from an Adabas data dictionary. You perform queries on data dictionary tables over the database link in the same way you query data dictionary tables in the Oracle Database. The gateway data dictionary is similar to the Oracle Database data dictionary in appearance and use.

This appendix contains the following sections:

- Data Dictionary Support
- Adabas Data Dictionary Descriptions

Data Dictionary Support

Data dictionary information is stored in either the gateway's advanced data dictionary (ADD) or the PREDICT table in the Adabas database. The views and tables that the gateway supports provide information on the following:

- The tables existing in the data source
- The columns for each table
- The primary key for each table
- The foreign keys for each table
- Which primary key each foreign key references
- The existing users and indexes

The following topics describe the data dictionary support for the Oracle Database Gateway.

Accessing the Gateway Data Dictionary

Accessing a definition of the ADD or the PREDICT table in the Adabas database is identical to accessing a data dictionary table or view definition in an Oracle database. You issue a SQL `SELECT` statement specifying a database link. The Oracle Database data dictionary view and column names are used to access the gateway data dictionary. Synonyms of supported views are also acceptable. For example, the following statement queries the data dictionary table `ALL_TABLES`:

```
SQL> SELECT * FROM ALL_TABLES@Adabas;
```

When a data dictionary access query is issued, the gateway:

1. Maps the requested table, view, or synonym to one or more ADD names.

2. Processes the query within the gateway.
3. May convert the retrieved data to give it the appearance of the Oracle Database data dictionary table.
4. Passes the data dictionary information to the Oracle Database.

Supported Views and Tables

The gateway supports the following views and tables:

- Supported Views and Tables
- ALL_CATALOG
- ALL_CONS_COLUMNS
- ALL_IND_COLUMNS
- ALL_OBJECTS
- ALL_TAB_COMMENTS
- ALL_USERS
- DBA_CATALOG
- DBA_OBJECTS
- DICTIONARY
- USER_CATALOG
- USER_CONS_COLUMNS
- USER_IND_COLUMNS
- USER_OBJECTS
- USER_TAB_COMMENTS
- USER_USERS
- ALL_COL_COMMENTS
- ALL_CONSTRAINTS
- ALL_INDEXES
- ALL_TAB_COLUMNS
- ALL_TABLES
- ALL_VIEWS
- DBA_TAB_COLUMNS
- DICT_COLUMNS
- DUAL
- USER_COL_COMMENTS
- USER_CONSTRAINTS
- USER_INDEXES
- USER_TAB_COLUMNS
- USER_TABLES
- USER_VIEWS

No other Oracle Database data dictionary tables or views are supported. If you use a view not on the list, you receive the Oracle Database error code for no more rows available.

If you want to select data dictionary views using `SELECT FROM DBA_*`, first connect as Oracle user `SYSTEM` or `SYS`. Otherwise, you receive the following error message:

```
ORA-28506: Parse error in data dictionary translation for %s stored in %s
```

Queries through the gateway of any data dictionary table or view beginning with ALL_ can returns rows from Adabas even when access privileges for those Adabas objects have not been granted. When querying an Oracle database with the Oracle data dictionary, rows are returned only for those objects you are permitted to access.

> **See Also:** *Oracle Database Error Messages* for more information on error messages.

Default Column Values

There is a minor difference between the gateway data dictionary and a typical Oracle Database data dictionary. The Oracle Database columns that are missing in an Adabas data dictionary table are filled with zeros, spaces, null values, or default values, depending on the column type.

Adabas Data Dictionary Descriptions

The gateway data dictionary tables and views provide the following information:

- Name, datatype, and width of each column
- The contents of columns with fixed values

In the descriptions that follow, the values in the Null? column may differ from the Oracle Database data dictionary tables and views. Any default value is shown to the right of an item.

> **Note:** In all data dictionary tables, the OWNER column has the value `ADABAS`.

Table C–1 describes the ALL_CATALOG table.

Table C–1 ALL_CATALOG

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_TYPE	-	VARCHAR2(11)	"TABLE"

Table C–2 describes the ALL_COL_COMMENTS table.

Table C–2 ALL_COL_COMMENTS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
TABLE_NAME	NOT NULL	VARCHAR2(30)	-

Table C–2 (Cont.) ALL_COL_COMMENTS

Name	Null?	Type	Value
COLUMN_NAME	NOT NULL	VARCHAR2(30)	-
COMMENTS	NOT NULL	CHAR(1)	-

Table C–3 describes the ALL_CONS_COLUMNS table.

Table C–3 ALL_CONS_COLUMNS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COLUMN_NAME	-	VARCHAR2(4000)	-
POSITION	-	FLOAT(49)	-

Table C–4 describes the ALL_CONSTRAINTS table.

Table C–4 ALL_CONSTRAINTS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)	-
CONSTRAINT_TYPE	-	VARCHAR2(1)	"R" or "P"
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
SEARCH_CONDITION	NOT NULL	CHAR(1)	-
R_OWNER	-	VARCHAR2(30)	-
R_CONSTRAINT_NAME	-	VARCHAR2(30)	-
DELETE_RULE	-	VARCHAR2(9)	-
STATUS	NOT NULL	CHAR(1)	-
DEFERRABLE	NOT NULL	CHAR(1)	-
DEFERRED	NOT NULL	CHAR(1)	-
VALIDATED	NOT NULL	CHAR(1)	-
GENERATED	NOT NULL	CHAR(1)	-
BAD	NOT NULL	CHAR(1)	-
RELY	NOT NULL	CHAR(1)	-
LAST_CHANGE	-	DATE	NULL

Table C–5 describes the ALL_IND-COLUMNS table.

Table C–5 ALL_IND_COLUMNS

Name	Null?	Type	Value
INDEX_OWNER	NOT NULL	VARCHAR2(30)	-
INDEX_NAME	NOT NULL	VARCHAR2(30)	-

Table C–5 (Cont.) ALL_IND_COLUMNS

Name	Null?	Type	Value
TABLE_OWNER	NOT NULL	VARCHAR2(30)	-
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COLUMN_NAME	-	VARCHAR2(4000)	-
COLUMN_POSITION	NOT NULL	FLOAT(49)	-
COLUMN_LENGTH	NOT NULL	FLOAT(49)	-
DESCEND	-	VARCHAR2(4)	NULL

Table C–6 describes the ALL_INDEXES table.

Table C–6 ALL_INDEXES

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
INDEX_NAME	NOT NULL	VARCHAR2(30)	-
INDEX_TYPE	NOT NULL	CHAR(1)	-
TABLE_OWNER	NOT NULL	VARCHAR2(30)	-
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_TYPE	NOT NULL	CHAR(5)	"TABLE"
UNIQUENESS	-	VARCHAR2(9)	"UNIQUE" or "NONUNIQUE"
COMPRESSION	NOT NULL	CHAR(1)	-
PREFIX_LENGTH	NOT NULL	NUMBER(10)	0
TABLESPACE_NAME	NOT NULL	CHAR(1)	-
INI_TRANS	NOT NULL	NUMBER(10)	0
MAX_TRANS	NOT NULL	NUMBER(10)	0
INITIAL_EXTENT	NOT NULL	NUMBER(10)	0
NEXT_EXTENT	NOT NULL	NUMBER(10)	0
MIN_EXTENTS	NOT NULL	NUMBER(10)	0
MAX_EXTENTS	NOT NULL	NUMBER(10)	0
PCT_INCREASE	NOT NULL	NUMBER(10)	0
PCT_THRESHOLD	NOT NULL	NUMBER(10)	0
INCLUDE_COLUMN	NOT NULL	NUMBER(10)	0
FREELISTS	NOT NULL	NUMBER(10)	0
FREELIST_GROUPS	NOT NULL	NUMBER(10)	0
PCT_FREE	NOT NULL	NUMBER(10)	0
LOGGING	NOT NULL	CHAR(1)	-
BLEVEL	NOT NULL	NUMBER(10)	0
LEAF_BLOCKS	NOT NULL	NUMBER(10)	0
DISTINCT_KEYS	-	FLOAT(49)	-

Table C–6 (Cont.) ALL_INDEXES

Name	Null?	Type	Value
AVG_LEAF_BLOCKS_PER_KEY	NOT NULL	NUMBER(10)	0
AVG_DATA_BLOCKS_PER_KEY	NOT NULL	NUMBER(10)	0
CLUSTERING_FACTOR	NOT NULL	NUMBER(10)	0
STATUS	NOT NULL	CHAR(1)	NULL
NUM_ROWS	NOT NULL	NUMBER(10)	0
SAMPLE_SIZE	NOT NULL	NUMBER(10)	0
LAST_ANALYZED	-	DATE	NULL
DEGREE	NOT NULL	CHAR(1)	NULL
INSTANCES	NOT NULL	CHAR(1)	NULL
PARTITIONED	NOT NULL	CHAR(1)	NULL
TEMPORARY	NOT NULL	CHAR(1)	-
GENERATED	NOT NULL	CHAR(1)	-
SECONDARY	NOT NULL	CHAR(1)	-
BUFFER_POOL	NOT NULL	CHAR(1)	-
USER_STATS	NOT NULL	CHAR(1)	-
DURATION	NOT NULL	CHAR(1)	-
PCT_DIRECT_ACCESS	NOT NULL	NUMBER(10)	0
ITYP_OWNER	NOT NULL	CHAR(1)	-
ITYP_NAME	NOT NULL	CHAR(1)	-
PARAMETERS	NOT NULL	CHAR(1)	-
GLOBAL_STATS	NOT NULL	CHAR(1)	-
DOMIDX_STATUS	NOT NULL	CHAR(1)	-
DOMIDX_OPSTATUS	NOT NULL	CHAR(1)	-
FUNCIDX_STATUS	NOT NULL	CHAR(1)	-

Table C–7 describes the ALL_OBJECTS table.

Table C–7 ALL_OBJECTS

Name	Null?	Type	Value
OWNER	-	VARCHAR2(30)	ADABAS
OBJECT_NAME	-	VARCHAR2(30)	-
SUBOBJECT_NAME	-	VARCHAR2(1)	NULL
OBJECT_ID	-	NUMBER	0
DATA_OBJECT_ID	-	NUMBER	0
OBJECT_TYPE	-	VARCHAR2(18)	"TABLE" or "INDEX"
CREATED	-	DATE	NULL
LAST_DDL_TIME	-	DATE	NULL

Table C–7 (Cont.) ALL_OBJECTS

Name	Null?	Type	Value
TIMESTAMP	-	VARCHAR2(1)	NULL
STATUS	-	VARCHAR2(1)	NULL
TEMPORARY	-	VARCHAR2(1)	NULL
GENERATED	-	VARCHAR2(1)	NULL
SECONDARY	-	VARCHAR2(1)	NULL

Table C–8 describes the ALL_TAB_COLUMNS table.

Table C–8 ALL_TAB_COLUMNS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COLUMN_NAME	NOT NULL	VARCHAR2(30)	-
DATA_TYPE	-	VARCHAR2(106)	-
DATA_TYPE_MOD	NOT NULL	CHAR(1)	-
DATA_TYPE_OWNER	NOT NULL	CHAR(1)	-
DATA_LENGTH	NOT NULL	FLOAT(49)	-
DATA_PRECISION	-	FLOAT(49)	-
DATA_SCALE	-	FLOAT(49)	-
NULLABLE	-	VARCHAR2(1)	"Y" or "N"
COLUMN_ID	NOT NULL	FLOAT(49)	-
DEFAULT_LENGTH	NOT NULL	NUMBER(10)	0
DATA_DEFAULT	NOT NULL	CHAR(1)	-
NUM_DISTINCT	NOT NULL	NUMBER(10)	0
LOW_VALUE	NOT NULL	NUMBER(10)	-
HIGH_VALUE	NOT NULL	NUMBER(10)	-
DENSITY	NOT NULL	NUMBER(10)	0
NUM_NULLS	NOT NULL	NUMBER(10)	0
NUM_BUCKETS	NOT NULL	NUMBER(10)	0
LAST_ANALYZED	-	DATE	NULL
SAMPLE_SIZE	NOT NULL	NUMBER(10)	0
CHARACTER_SET_NAME	NOT NULL	CHAR(1)	-
CHAR_COL_DEC_LENGTH	NOT NULL	NUMBER(10)	0
GLOBAL_STATS	NOT NULL	CHAR(1)	-
USER_STATS	NOT NULL	CHAR(1)	-
AVG_COL_LEN	NOT NULL	NUMBER(10)	0

Table C–9 describes the ALL_TAB_COMMENTS table.

Table C–9 ALL_TAB_COMMENTS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_TYPE	-	VARCHAR2(11)	"TABLE"
COMMENTS	NOT NULL	CHAR(1)	-

Table C–10 describes the ALL_TABLES table.

Table C–10 ALL_TABLES

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
TABLESPACE_NAME	NOT NULL	CHAR(1)	-
CLUSTER_NAME	NOT NULL	CHAR(1)	-
IOT_NAME	NOT NULL	CHAR(1)	-
PCT_FREE	NOT NULL	NUMBER(10)	0
PCT_USED	NOT NULL	NUMBER(10)	0
INI_TRANS	NOT NULL	NUMBER(10)	0
MAX_TRANS	NOT NULL	NUMBER(10)	0
INITIAL_EXTENT	NOT NULL	NUMBER(10)	0
NEXT_EXTENT	NOT NULL	NUMBER(10)	0
MIN_EXTENTS	NOT NULL	NUMBER(10)	0
MAX_EXTENTS	NOT NULL	NUMBER(10)	0
PCT_INCREASE	NOT NULL	NUMBER(10)	0
FREELISTS	NOT NULL	NUMBER(10)	0
FREELIST_GROUPS	NOT NULL	NUMBER(10)	0
LOGGING	NOT NULL	CHAR(1)	-
BACKED_UP	NOT NULL	CHAR(1)	-
NUM_ROWS	-	FLOAT(49)	-
BLOCKS	-	FLOAT(49)	-
EMPTY_BLOCKS	NOT NULL	NUMBER(10)	0
AVG_SPACE	NOT NULL	NUMBER(10)	0
CHAIN_CNT	NOT NULL	NUMBER(10)	0
AVG_ROW_LEN	NOT NULL	NUMBER(10)	0
AVG_SPACE_FREELIST_BLOCKS	NOT NULL	NUMBER(10)	0
NUM_FREELIST_BLOCKS	NOT NULL	NUMBER(10)	0
DEGREE	NOT NULL	CHAR(1)	-
INSTANCES	NOT NULL	CHAR(1)	-
CACHE	NOT NULL	CHAR(1)	-

Table C–10 (Cont.) ALL_TABLES

Name	Null?	Type	Value
TABLE_LOCK	NOT NULL	CHAR(1)	-
SAMPLE_SIZE	NOT NULL	NUMBER (10)	0
LAST_ANALYZED	-	DATE	NULL
PARTITIONED	NOT NULL	CHAR(1)	-
IOT_TYPE	NOT NULL	CHAR(1)	-
TEMPORARY	NOT NULL	CHAR(1)	-
SECONDARY	NOT NULL	CHAR(1)	-
NESTED	NOT NULL	CHAR(1)	-
BUFFER_POOL	NOT NULL	CHAR(1)	-
ROW_MOVEMENT	NOT NULL	CHAR(1)	-
GLOBAL_STATS	NOT NULL	CHAR(1)	-
USER_STATS	NOT NULL	CHAR(1)	-
DURATION	NOT NULL	CHAR(1)	-
SKIP_CORRUPT	NOT NULL	CHAR(1)	-
MONITORING	NOT NULL	CHAR(1)	-

Table C–11 describes the ALL_USERS table.

Table C–11 ALL_USERS

Name	Null?	Type	Value
USERNAME	NOT NULL	VARCHAR2(30)	-
USER_ID	-	NUMBER	0
CREATED	-	DATE	NULL

Table C–12 describes the ALL_VIEWS table.

Table C–12 ALL_VIEWS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
VIEW_NAME	NOT NULL	VARCHAR2(30)	-
TEXT_LENGTH	NOT NULL	NUMBER(10)	0
TEXT	NOT NULL	CHAR(1)	-
TYPE_TEXT_LENGTH	NOT NULL	NUMBER(10)	0
TYPE_TEXT	NOT NULL	CHAR(1)	-
OID_TEXT_LENGTH	NOT NULL	NUMBER(10)	0
OID_TEXT	NOT NULL	CHAR(1)	-
VIEW_TYPE_OWNER	NOT NULL	CHAR(1)	-
VIEW_TYPE	NOT NULL	CHAR(1)	-

Table C–13 describes the DBA_CATALOG table.

Table C–13 DBA_CATALOG

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_TYPE	-	VARCHAR2(11)	"TABLE"

Table C–14 describes the DBA_OBJECTS table.

Table C–14 DBA_OBJECTS

Name	Null?	Type	Value
OWNER	-	VARCHAR2(30)	ADABAS
OBJECT_NAME	-	VARCHAR2(128)	-
SUBOBJECT_NAME	-	VARCHAR2(1)	NULL
OBJECT_ID	-	NUMBER	0
DATA_OBJECT_ID	-	NUMBER	0
OBJECT_TYPE	-	VARCHAR2(18)	"TABLE" or "INDEX"
CREATED	-	DATE	NULL
LAST_DDL_TIME	-	DATE	NULL
TIMESTAMP	-	VARCHAR2(1)	NULL
STATUS	-	VARCHAR2(1)	NULL
TEMPORARY	-	VARCHAR2(1)	NULL
GENERATED	-	VARCHAR2(1)	NULL
SECONDARY	-	VARCHAR2(1)	NULL

Table C–15 describes the DBA_TAB_COLUMNS table.

Table C–15 DBA_TAB_COLUMNS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COLUMN_NAME	NOT NULL	VARCHAR2(30)	-
DATA_TYPE	-	VARCHAR2(106)	-
DATA_TYPE_MOD	NOT NULL	CHAR(1)	-
DATA_TYPE_OWNER	NOT NULL	CHAR(1)	-
DATA_LENGTH	NOT NULL	FLOAT(49)	-
DATA_PRECISION	-	FLOAT(49)	-
DATA_SCALE	-	FLOAT(49)	-
NULLABLE	-	VARCHAR2(1)	"Y" or "N"
COLUMN_ID	NOT NULL	FLOAT(49)	-

Table C–15 (Cont.) DBA_TAB_COLUMNS

Name	Null?	Type	Value
DEFAULT_LENGTH	NOT NULL	NUMBER(10)	0
DATA_DEFAULT	NOT NULL	CHAR(1)	-
NUM_DISTINCT	NOT NULL	NUMBER(10)	0
LOW_VALUE	NOT NULL	NUMBER(10)	-
HIGH_VALUE	NOT NULL	NUMBER(10)	-
DENSITY	NOT NULL	NUMBER(10)	0
NUM_NULLS	NOT NULL	NUMBER(10)	0
NUM_BUCKETS	NOT NULL	NUMBER(10)	0
LAST_ANALYZED	-	DATE	NULL
SAMPLE_SIZE	NOT NULL	NUMBER(10)	0
CHARACTER_SET_NAME	NOT NULL	CHAR(1)	-
CHAR_COL_DEC_LENGTH	NOT NULL	NUMBER(10)	0
GLOBAL_STATS	NOT NULL	CHAR(1)	-
USER_STATS	NOT NULL	CHAR(1)	-
AVG_COL_LEN	NOT NULL	NUMBER(10)	0

Table C–16 describes the DBA_COLUMNS table.

Table C–16 DICT_COLUMNS

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COLUMN_NAME	NOT NULL	VARCHAR2(30)	-
COMMENTS	NOT NULL	CHAR(1)	-

Table C–17 describes the DICTIONARY table.

Table C–17 DICTIONARY

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COMMENTS	NOT NULL	CHAR(1)	-

Table C–18 describes the DUAL table.

Table C–18 DUAL

Name	Null?	Type	Value
DUMMY	NOT NULL	VARCHAR2(1)	-

Table C–19 describes the USER_CATALOG table.

Table C–19 USER_CATALOG

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_TYPE	-	VARCHAR2(11)	"TABLE"

Table C–20 describes the USER_COL_COMMENTS table.

Table C–20 USER_COL_COMMENTS

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COLUMN_NAME	NOT NULL	VARCHAR2(30)	-
COMMENTS	NOT NULL	CHAR(1)	-

Table C–21 describes the USER_CONS_COLUMNS table.

Table C–21 USER_CONS_COLUMNS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COLUMN_NAME	-	VARCHAR2(4000)	-
POSITION	-	FLOAT(49)	-

Table C–22 describes the USER_CONSTRAINTS table.

Table C–22 USER_CONSTRAINTS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	ADABAS
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)	-
CONSTRAINT_TYPE	-	VARCHAR2(1)	"R" or "P"
TABLE_NAME	-	VARCHAR2(30)	-
SEARCH_CONDITION	NOT NULL	CHAR(1)	-
R_OWNER	NOT NULL	VARCHAR2(1)	-
R_CONSTRAINT_NAME	-	VARCHAR2(30)	-
DELETE_RULE	-	VARCHAR2(9)	NULL
STATUS	NOT NULL	CHAR(1)	-
DEFERRABLE	NOT NULL	CHAR(1)	-
DEFERRED	NOT NULL	CHAR(1)	-
VALIDATED	NOT NULL	CHAR(1)	-
GENERATED	NOT NULL	CHAR(1)	-
BAD	NOT NULL	CHAR(1)	-
RELY	NOT NULL	CHAR(1)	-

Table C–22 (Cont.) USER_CONSTRAINTS

Name	Null?	Type	Value
LAST_CHANGE	-	DATE	NULL

Table C–23 describes the USER_IND_COLUMNS table.

Table C–23 USER_IND_COLUMNS

Name	Null?	Type	Value
INDEX_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COLUMN_NAME	-	VARCHAR2(4000)	-
COLUMN_POSITION	NOT NULL	FLOAT(49)	-
COLUMN_LENGTH	NOT NULL	FLOAT(49)	-
DESCEND	-	VARCHAR2(4)	"DESC" or "ASC"

Table C–24 describes the USER_INDEXES table.

Table C–24 USER_INDEXES

Name	Null?	Type	Value
INDEX_NAME	NOT NULL	VARCHAR2(30)	-
INDEX_TYPE	NOT NULL	CHAR(1)	-
TABLE_OWNER	NOT NULL	VARCHAR2(30)	-
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_TYPE	NOT NULL	CHAR(5)	"TABLE"
UNIQUENESS	-	VARCHAR2(9)	"UNIQUE" or "NONUNIQUE"
COMPRESSION	NOT NULL	CHAR(1)	-
PREFIX_LENGTH	NOT NULL	NUMBER(10)	0
TABLESPACE_NAME	NOT NULL	CHAR(1)	-
INI_TRANS	NOT NULL	NUMBER(10)	0
MAX_TRANS	NOT NULL	NUMBER(10)	0
INITIAL_EXTENT	NOT NULL	NUMBER(10)	0
NEXT_EXTENT	NOT NULL	NUMBER(10)	0
MIN_EXTENTS	NOT NULL	NUMBER(10)	0
MAX_EXTENTS	NOT NULL	NUMBER(10)	0
PCT_INCREASE	NOT NULL	NUMBER(10)	0
PCT_THRESHOLD	NOT NULL	NUMBER(10)	0
INCLUDE_COLUMN	NOT NULL	NUMBER(10)	0
FREELISTS	NOT NULL	NUMBER(10)	0
FREELIST_GROUPS	NOT NULL	NUMBER(10)	0
PCT_FREE	NOT NULL	NUMBER(10)	0

Table C–24 (Cont.) USER_INDEXES

Name	Null?	Type	Value
LOGGING	NOT NULL	CHAR(1)	-
BLEVEL	NOT NULL	NUMBER(10)	0
LEAF_BLOCKS	NOT NULL	NUMBER(10)	0
DISTINCT_KEYS	-	FLOAT(49)	-
AVG_LEAF_BLOCKS_PER_KEY	NOT NULL	NUMBER(10)	0
AVG_DATA_BLOCKS_PER_KEY	NOT NULL	NUMBER(10)	0
CLUSTERING_FACTOR	NOT NULL	NUMBER(10)	0
STATUS	NOT NULL	CHAR(1)	-
NUM_ROWS	NOT NULL	NUMBER(10)	0
SAMPLE_SIZE	NOT NULL	NUMBER(10)	0
LAST_ANALYZED	-	DATE	NULL
DEGREE	NOT NULL	CHAR(1)	-
INSTANCES	NOT NULL	CHAR(1)	-
PARTITIONED	NOT NULL	CHAR(1)	-
TEMPORARY	NOT NULL	CHAR(1)	-
GENERATED	NOT NULL	CHAR(1)	-
SECONDARY	NOT NULL	CHAR(1)	-
BUFFER_POOL	NOT NULL	CHAR(1)	-
USER_STATS	NOT NULL	CHAR(1)	-
DURATION	NOT NULL	CHAR(1)	-
PCT_DIRECT_ACCESS	NOT NULL	NUMBER(10)	0
ITYP_OWNER	NOT NULL	CHAR(1)	-
ITYP_NAME	NOT NULL	CHAR(1)	-
PARAMETERS	NOT NULL	CHAR(1)	-
GLOBAL_STATS	NOT NULL	CHAR(1)	-
DOMIDX_STATUS	NOT NULL	CHAR(1)	-
DOMIDX_OPSTATUS	NOT NULL	CHAR(1)	-
FUNCIDX_STATUS	NOT NULL	CHAR(1)	-

Table C–25 describes the USER_OBJECTS table.

Table C–25 USER_OBJECTS

Name	Null?	Type	Value
OBJECT_NAME	-	VARCHAR2(128)	-
SUBOBJECT_NAME	-	VARCHAR2(1)	NULL
OBJECT_ID	-	NUMBER	0
DATA_OBJECT_ID	-	NUMBER	0

Table C–25 (Cont.) USER_OBJECTS

Name	Null?	Type	Value
OBJECT_TYPE	-	VARCHAR2(18)	"TABLE" or "INDEX"
CREATED	-	DATE	NULL
LAST_DDL_TIME	-	DATE	NULL
TIMESTAMP	-	VARCHAR2(1)	NULL
STATUS	-	VARCHAR2(1)	NULL
TEMPORARY	-	VARCHAR2(1)	NULL
GENERATED	-	VARCHAR2(1)	NULL
SECONDARY	-	VARCHAR2(1)	NULL

Table C–26 describes the USER_TAB_COLUMNS table.

Table C–26 USER_TAB_COLUMNS

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
COLUMN_NAME	NOT NULL	VARCHAR2(30)	-
DATA_TYPE	-	VARCHAR2(106)	-
DATA_TYPE_MOD	NOT NULL	CHAR(1)	-
DATA_TYPE_OWNER	NOT NULL	CHAR(1)	-
DATA_LENGTH	NOT NULL	FLOAT(49)	-
DATA_PRECISION	-	FLOAT(49)	-
DATA_SCALE	-	FLOAT(49)	-
NULLABLE	-	VARCHAR2(1)	"Y" or "N"
COLUMN_ID	NOT NULL	FLOAT(49)	-
DEFAULT_LENGTH	NOT NULL	NUMBER(10)	0
DATA_DEFAULT	NOT NULL	CHAR(1)	-
NUM_DISTINCT	NOT NULL	NUMBER(10)	0
LOW_VALUE	NOT NULL	NUMBER(10)	0
HIGH_VALUE	NOT NULL	NUMBER(10)	0
DENSITY	NOT NULL	NUMBER(10)	0
NUM_NULLS	NOT NULL	NUMBER(10)	0
NUM_BUCKETS	NOT NULL	NUMBER(10)	0
LAST_ANALYZED	-	DATE	NULL
SAMPLE_SIZE	NOT NULL	NUMBER(10)	0
CHARACTER_SET_NAME	NOT NULL	CHAR(1)	-
CHAR_COL_DECL_LENGTH	NOT NULL	NUMBER(10)	0
GLOBAL_STATS	NOT NULL	CHAR(1)	-
USER_STATS	NOT NULL	CHAR(1)	-

Table C–26 (Cont.) USER_TAB_COLUMNS

Name	Null?	Type	Value
AVG_COL_LEN	NOT NULL	NUMBER(10)	0

Table C–27 describes the USER_TAB_COMMENTS table.

Table C–27 USER_TAB_COMMENTS

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
TABLE_TYPE	-	VARCHAR2(11)	"TABLE"
COMMENTS	NOT NULL	CHAR(1)	-

Table C–28 describes the USER_TABLES table.

Table C–28 USER_TABLES

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2(30)	-
TABLESPACE_NAME	NOT NULL	CHAR(1)	-
CLUSTER_NAME	NOT NULL	CHAR(1)	-
IOT_NAME	NOT NULL	CHAR(1)	-
PCT_FREE	NOT NULL	NUMBER(10)	0
PCT_USED	NOT NULL	NUMBER(10)	0
INI_TRANS	NOT NULL	NUMBER(10)	0
MAX_TRANS	NOT NULL	NUMBER(10)	0
INITIAL_EXTENT	NOT NULL	NUMBER(10)	0
NEXT_EXTENT	NOT NULL	NUMBER(10)	0
MIN_EXTENTS	NOT NULL	NUMBER(10)	0
MAX_EXTENTS	NOT NULL	NUMBER(10)	0
PCT_INCREASE	NOT NULL	NUMBER(10)	0
FREELISTS	NOT NULL	NUMBER(10)	0
FREELIST_GROUPS	NOT NULL	NUMBER(10)	0
LOGGING	NOT NULL	CHAR(1)	-
BACKED_UP	NOT NULL	CHAR(1)	-
NUM_ROWS	-	FLOAT(49)	-
BLOCKS	-	FLOAT(49)	-
EMPTY_BLOCKS	NOT NULL	NUMBER(10)	0
AVG_SPACE	NOT NULL	NUMBER(10)	0
CHAIN_CNT	NOT NULL	NUMBER(10)	0
AVG_ROW_LEN	NOT NULL	NUMBER(10)	0
AVG_SPACE_FREELIST_BLOCKS	NOT NULL	NUMBER(10)	0
NUM_FREELIST_BLOCKS	NOT NULL	NUMBER(10)	0

Table C–28 (Cont.) USER_TABLES

Name	Null?	Type	Value
DEGREE	NOT NULL	CHAR(1)	-
INSTANCES	NOT NULL	CHAR(1)	-
CACHE	NOT NULL	CHAR(1)	-
TABLE_LOCK	NOT NULL	CHAR(1)	-
SAMPLE_SIZE	NOT NULL	NUMBER(10)	0
LAST_ANALYZED	-	DATE	NULL
PARTITIONED	NOT NULL	CHAR(1)	-
IOT_TYPE	NOT NULL	CHAR(1)	-
TEMPORARY	NOT NULL	CHAR(1)	-
SECONDARY	NOT NULL	CHAR(1)	-
NESTED	NOT NULL	CHAR(1)	-
BUFFER_POOL	NOT NULL	CHAR(1)	-
ROW_MOVEMENT	NOT NULL	CHAR(1)	-
GLOBAL_STATS	NOT NULL	CHAR(1)	-
USER_STATS	NOT NULL	CHAR(1)	-
DURATION	NOT NULL	CHAR(1)	-
SKIP_CORRUPT	NOT NULL	CHAR(1)	-
MONITORING	NOT NULL	CHAR(1)	-

Table C–29 describes the USER_USERS table.

Table C–29 USER_USERS

Name	Null?	Type	Value
USERNAME	NOT NULL	VARCHAR2(30)	-
USER_ID	-	NUMBER	0
ACCOUNT_STATUS	-	VARCHAR2(4)	"OPEN"
LOCK_DATE	-	DATE	NULL
EXPIRY_DATE	-	DATE	NULL
DEFAULT_TABLESPACE	-	VARCHAR2(1)	NULL
TEMPORARY_TABLESPACE	-	VARCHAR2(1)	NULL
CREATED	-	DATE	NULL
INITIAL_RSRC_CONSUMER_GROUP	-	VARCHAR2(1)	NULL
EXTERNAL_NAME	-	VARCHAR2(1)	NULL

Table C–30 describes the USER_VIEWS table.

Table C–30 USER_VIEWS

Name	Null?	Type	Value
VIEW_NAME	NOT NULL	VARCHAR2(30)	-

Table C–30 (Cont.) USER_VIEWS

Name	Null?	Type	Value
TEXT_LENGTH	NOT NULL	NUMBER(10)	0
TEXT	NOT NULL	CHAR(1)	-
TYPE_TEXT_LENGTH	NOT NULL	NUMBER(10)	0
TYPE_TEXT	NOT NULL	CHAR(1)	-
OID_TEXT_LENGTH	NOT NULL	NUMBER(10)	0
OID_TEXT	NOT NULL	CHAR(1)	-
VIEW_TYPE_OWNER	NOT NULL	CHAR(1)	-
VIEW_TYPE	NOT NULL	CHAR(1)	-

D

Globalization Support

The main aspect of the Globalization Support in Oracle Connect for IMS, VSAM, and Adabas Gateways is the recognition of the different characters associated with a language and the way they are encoded in various operating systems and data sources. For each supported language, a special definition file called a character set file is supplied where all the language related information is stored. For complex languages such as Chinese, Japanese, and Korean, a special library is also provided where specific conversion rules are implemented.

As a distributed product that accesses heterogeneous data sources on varied platforms, Oracle Connect for IMS, VSAM, and Adabas Gateways offers seamless conversion of text between the different character encodings used on the different platforms. Examples of such automatic conversion include:

- Conversion between ASCII based encoding on open systems and EBCDIC based encoding on IBM mainframes and AS/400 machines
- Conversions to and from Unicode for databases that store data in Unicode
- Conversions between different encodings of the same language used on different platforms
- Conversions of legacy data stored using old character encodings (such as 7-bit encoding) into the current platform encoding standard

Getting this kind of seamless Globalization Support requires the proper setting of the character set definitions according to the kind of encoding in use in the various data sources and platforms.

This section discusses the different encoding schemes in use, the character set definitions required and other Globalization Support related aspects, and contains information on the following topics:

- Character Set Terminology
- Globalization Support Settings

Character Set Terminology

The following terminology is used to describe character sets.

Single-Byte Character Sets

In a single-byte character set, each character is represented by a single-byte value, that is, a number between 1 and 255, inclusive. Single-byte character sets are typical of Western languages. For example, in the ISO-8859-1 (Latin) character set, the character 'A' is represented by the single byte value of 65, whereas in the US-EBCDIC character

set (or in the IBM-037 character set), the same character is represented by the single-byte value of 193.

Multibyte Character Sets

In a multibyte character set, some or all of the characters are represented by more than one byte value. Multibyte character sets are typical in complex languages such as Chinese, Japanese and Korean.

Unicode Character Sets

Unicode is a universal numbering of all known characters, with each character identified by a unique number - its codepoint. Unicode has several encoding schemes, of which Oracle Application Development Framework Controller API Reference supports UTF-8 and, to a lesser extent, UCS-2.

Since the product uses 8-bit characters, the only Unicode encoding that qualifies as a 'character set' is the UTF-8 encoding. The product supports UCS-2 in its data sources (through special Unicode data types).

Customized Character Sets

The Globalization Support of Oracle Connect can be customized to add new languages and character sets not currently supported as well as to introduce special conversion cases. The customization involves editing special character set source files and building .cp files from them using the NAV_UTIL program.

Globalization Support Settings

The minimal globalization Support configuration amounts to adding the HS_ LANGUAGE parameter to the HS initialization parameter file and telling the product what national language is in use.

For information on how to add the parameter to the HS initialization parameter file, see *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for Microsoft Windows* or *Oracle Database Gateway for IMS, VSAM, and Adabas Installation and Configuration Guide for AIX 5L Based Systems (64-Bit), HP-UX Itanium, Solaris Operating System (SPARC 64-Bit), Linux x86, and Linux x86-64*.

To set the language in Studio

1. In the Oracle Studio for IMS, VSAM, and Adabas Gateways Design perspective, open the machine for which you want to set the language.
2. Expand the Bindings and right-click the **NAV** binding.
3. Select **Edit Binding**.
4. Open the **Misc** category and fill in the language parameter with the desired language code from Globalization Support Language Codes.
5. Save the change. New servers will use the language selected.

When a language is selected, a default character set is automatically used based on the language and the platform. Table D–1 summarizes the languages, their codes, and their character sets.

Table D–1 Globalization Support Language Codes

EBCDIC CP Name	Description	Base ASCII CP	Multibyte
AR8EBCDIC420	Arabic bilingual	AR8ISO8859P6	

Table D–1 (Cont.) Globalization Support Language Codes

EBCDIC CP Name	Description	Base ASCII CP	Multibyte
AR8EBCDICX	Arabic + Latin	AR8ISO8859P6	
BLT8EBCDIC1112	Baltic multilingual	BLT8ISO8859P13	
CL8EBCDIC1025	Cyrillic multilingual	CL8ISO8859P5	
CL8EBCDIC1158	Cyrillic Ukraine + Euro	CLMSWIN1251	
D8EBCDIC1141	Austria - Germany + Euro	WE8ISO8859P15	
D8EBCDIC273	Germany - Austria	WE8ISO8859P1	
DK8EBCDIC1142	Denmark - Norway + Euro	WE8ISO8859P15	
DK8EBCDIC277	Denmark - Norway	NE8ISO8859P10	
EE8EBCDIC870	Latin 2 multilingual	EE8ISO8859P2	
EL8EBCDIC423	Greece	EL8ISO8859P7	
EL8EBCDIC875	Greece	EL8ISO8859P7	
EL8EBCDIC875R	Greece	EL8ISO8859P7	
F8EBCDIC1147	France + Euro	WE8ISO8859P15	
F8EBCDIC297	France	WE8ISO8859P1	
I8EBCDIC1144	Italy + Euro	WE8ISO8859P15	
I8EBCDIC280	Italy	WE8ISO8859P1	
IW8EBCDIC1086	Hebrew	IW8ISO8859P8	
IW8EBCDIC424	Hebrew	IW8ISO8859P8	
JA16DBCS	Japan	JA16SJIS	Yes
JA16EBCDIC930	Japan	JA16SJIS	Yes
KO16DBCS	Korea	KO16KSC5601	Yes
S8EBCDIC1143	Finland - Sweden + Euro	WE8ISO8859P15	
S8EBCDIC278	Finland - Sweden	WE8ISO8859P1	
TH8TISEBCDIC	Thai IS 620-2533 EBCDIC 8-bit	TH8TISASCII	
TR8EBCDIC1026	Turkey	WE8ISO8859P9	
WE8EBCDIC1047	Latin 1	WE8ISO8859P1	
WE8EBCDIC1140	USA, Canada + Euro	WE8ISO8859P15	
WE8EBCDIC1145	Spanish + Euro	WE8ISO8859P15	
WE8EBCDIC1146	UK + Euro	WE8ISO8859P15	
WE8EBCDIC1148	International ECECP + Euro	WE8ISO8859P15	
WE8EBCDIC1148	Western Europe + Euro	WE8ISO8859P15	
WE8EBCDIC284	Spanish	WE8ISO8859P1	
WE8EBCDIC285	UK	WE8ISO8859P1	
WE8EBCDIC37	USA + Canada	WE8ISO8859P1	
WE8EBCDIC37	Canadian French	WE8ISO8859P1	
WE8EBCDIC500	Western Europe	WE8ISO8859P1	
WE8EBCDIC871	Iceland	NE8ISO8859P10	

Table D–1 (Cont.) Globalization Support Language Codes

EBCDIC CP Name	Description	Base ASCII CP	Multibyte
WE8EBCDIC924	Latin 9	WE8ISO8859P9	
ZHS16DBCS	Simplified Chinese	ZHS16CGB231280	Yes
ZHT16DBCS	Traditional Chinese	ZHT16BIG5	Yes

Index

A

accessing Adabas data, 2-1
Adabas data, accessing, 2-1
Adabas, exporting metadata, 2-1
ADD
  atomic data types, A-1
  scaled data types, A-2
advanced data dictionary
  See ADD
ALTER statement, B-1
arithmetic operators, B-2
array handling
  normalized tables, 3-5
  overview, 3-1
  sequential flattening, 3-7
  setting default, 4-1
  setting policy, 4-1
  table settings, 4-2

C

case sensitivity, databases, 2-3
commands
  COMMIT, 2-7
  ROLLBACK, 2-7
commands, passing to database, 2-8
COMMIT command, restrictions, 2-7
comparison operators, B-3
CONCAT operator, 2-7
conversion, data types, 2-4, A-1
CREATE statement, B-1
cursor loops, restrictions, 2-7

D

data definition language, B-1
data dictionary views, 2-8
data dictionary, views, C-2
data types
  ADD atomic, A-1
  ADD scaled, A-2
  conversion, 2-4, A-1
  FLOAT, 2-8
  VARBINARY, 2-8
data, accessing, 2-1
data, non-relational, 3-1
databases, passing commands, 2-8
DDL
  See data definition language
default, array handling, 4-1
DELETE statement, B-1, B-2
DROP statement, B-1

E

encrypted format login, 2-8
errors
  ORA-02070, 2-7, B-2
  ORA-1002, 2-7
  ORA-28506, C-3
  ORA-6550, 2-7
exporting Adabas metadata, 2-1

F

FLOAT data type, 2-8
format login, encrypted, 2-8
functions, B-3
functions, unsupported, 2-6

G

gateways
  data dictionary tables, C-1
  functions, B-1
  pass-through feature, 2-3
  restrictions, 2-5
  SQL restrictions, 2-6
  SQL syntax, 2-2, B-1
GRANT statement, B-1
group functions, B-3

H

handling arrays
  normalized tables, 3-5
  overview, 3-1
  sequential flattening, 3-7
hexadecimal notation, 2-4

I

INSERT statement, B-1, B-2

L

locking, database, 2-5
login, encrypted format, 2-8

M

metadata
  exporting, 2-1
  representing, 3-1

N

naming rules, objects, 2-3
non-relational data, normalizing, 3-1
normalizing non-relation data, 3-1
notation, hexadecimal, 2-4

O

objects, naming rules, 2-3
ORA-02070 error, 2-7, B-2
ORA-1002 error, 2-7
ORA-28506 error, C-3
ORA-6550 error, 2-7

P

passing commands to database, 2-8
pass-through feature, 2-3
PL/SQL, 2-8
policy, array handling, 4-1
Predict metatada, 2-1

R

representing metadata, 3-1
restrictions
  COMMIT command, 2-7
  cursor loops, 2-7
  gateways, 2-5
  ROLLBACK command, 2-7
  SQL, 2-6
ROLLBACK command, restrictions, 2-7
ROWID, 2-6

S

savepoint support, 2-7
SELECT statement, B-1, C-1
sequential flattening, 3-7
sequential view, 3-7
single table, 3-7
SQL execution, 2-4
SQL functions, unsupported, 2-6
SQL support, 2-2
SQL syntax, supported, 2-2, B-1
statements
  DELETE, B-1
  DROP, B-1
  GRANT, B-1
  INSERT, B-1, B-2
  SELECT, B-1
  UPDATE, B-2
string functions, B-3
support, SQL, 2-2

T

transaction, 2-7
TRUNCATE statement, B-1

U

unsupported SQL functions, 2-6
UPDATE statement, B-2

V

VARBINARY data type, 2-8
views, data dictionary, 2-8