Oracle® Database Gateway for IMS

User's Guide

11*g* Release 1 (11.1)

B31053-01

July 2007

ORACLE®

Oracle Database Gateway for IMS User's Guide, 11*g* Release 1 (11.1)

B31053-01

Primary Author: Jeanne Wiegelmann

Contributing Author: Maitreyee Chaliha, Sami Zeitoun, Oussama Mkaabal

Contributor: Vira Goorah, Peter Wong

Contents

Preface

This manual describes the Oracle Database Gateway for IMS, which enables Oracle client applications to access IMS 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 IMS
- Diagnosing gateway errors
- Using the gateway to access IMS 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, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. 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.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, see the following documents in the Oracle Database Gateway 11*g* Release 1 (11.1) 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 Administrator'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.
`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

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 IMS.

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 Administrator's Guide* for more information about Heterogeneous Services.

Oracle Database Gateways

Oracle Database Gateway for IMS allows Oracle client applications to access IMS 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 IMS data source as if the data was stored in an Oracle table. A single SQL statement can access data residing in Oracle and IMS 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 IMS files, the physical location of the data is transparent to the client application. This allows future migration of data from IMS 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 IMS 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

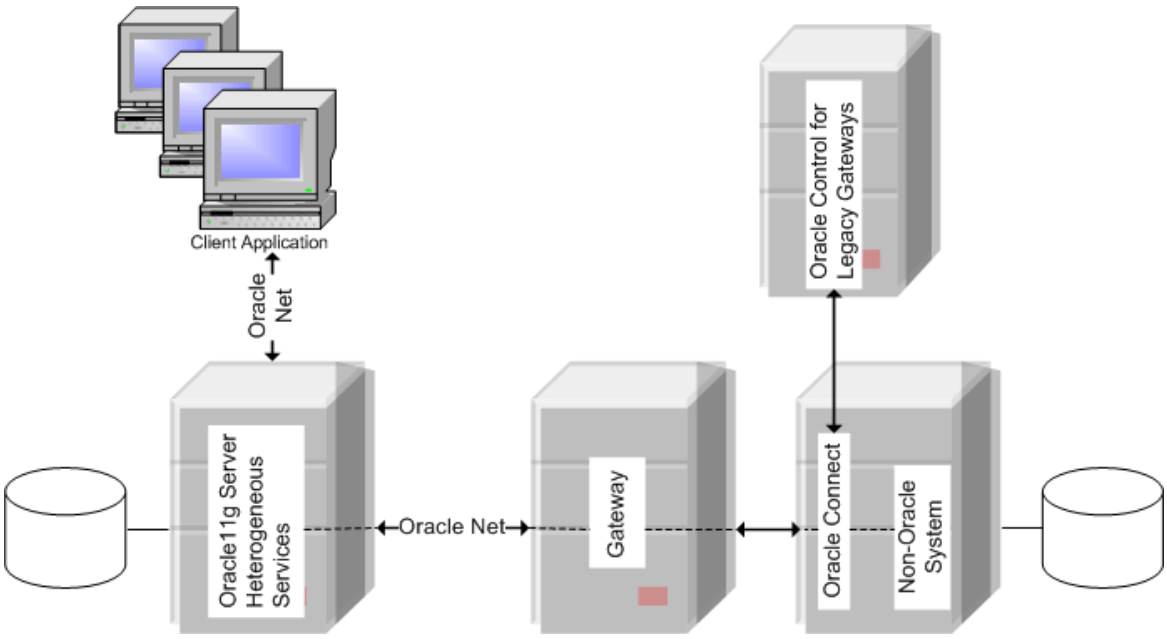
The gateway is invoked by the listener. The gateway is not multi-threaded and cannot support shared database links. Each gateway session spawns a separate gateway process, and connections cannot be shared.

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 IMS data source, as shown in Figure 1–1.

> **Note:** The non-Oracle system in Figure 1–1 and Figure 1–2 represents the IMS 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 IMS resides on a second system.
- The IMS 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 IMS 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.

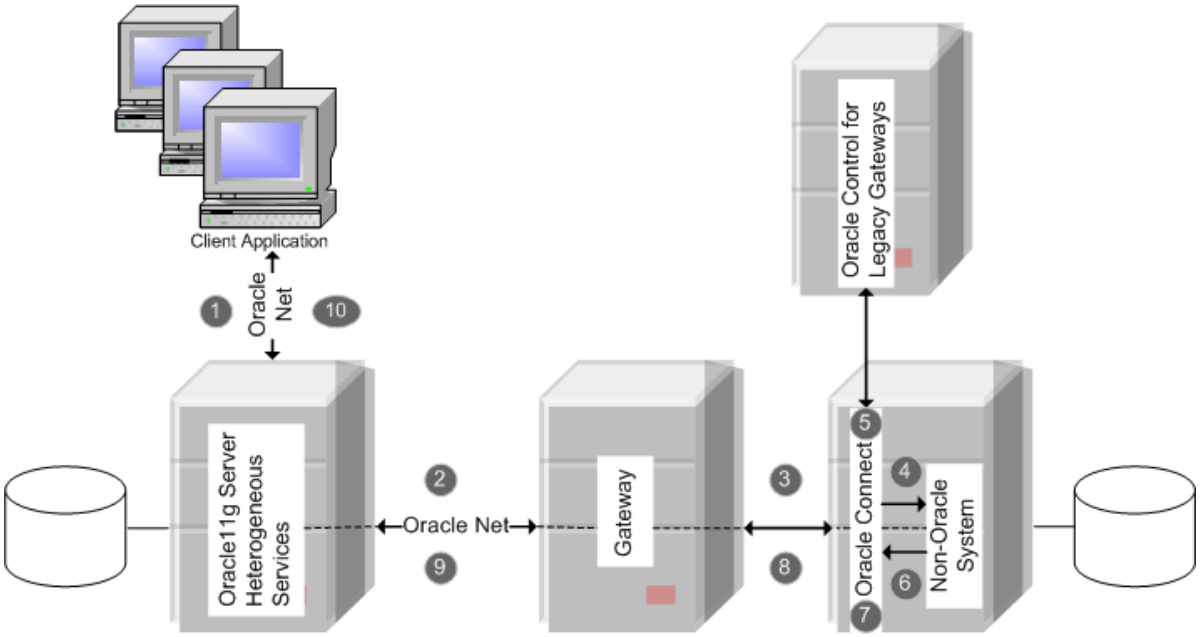
The Oracle database server and the gateway work together to present the appearance of a single Oracle database to the client. All data accessed by the client appears to reside in a single Oracle database. The client application sends a request to the Oracle database server, and the Oracle database server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

For the first transaction in a session, the gateway logs into the IMS data source using a username and password that is valid in the respective data source. The gateway converts the SQL statement to a native IMS statement, and the IMS 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 IMS 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 IMS 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 IMS 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 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 UNIX*

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 UNIX*

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 UNIX*

5. Set up the IMS 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 UNIX*

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 UNIX*

7. Install Oracle Database Gateway for IMS

   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 UNIX*

8. Configure Oracle Database Gateway for IMS

   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 UNIX*

2

Gateway Features and Restrictions

After the gateway is installed and configured, you can use the gateway to access IMS data, perform distributed queries, and copy data.

This chapter contains the following sections:

Accessing IMS/DB Data

Oracle Connect supports SQL data access to IMS/DB data in the following three IMS/DB environments:

- IMS-DLI: Batch access. The Oracle Connect server issues direct DLI commands to retrieve data as a standalone batch program. This means that the database is accessed from the Oracle Connect-started task without going through any of the IMS control regions. This access method is suited to nightly processing, such as bulk loading when the IMS control region is down. It is usually not suited for general multi-user data access.
- IMS-DBCTL: This data source is suited to users employing CICS as their primary application platform for accessing IMS data. All Oracle Connect servers communicate with an Oracle Connect-supplied CICS program. This program accepts requests for scheduling PSBs and retrieving data through DBCTL services.
- IMS-DBDC: This data source is suited to users employing IMS/TM as their primary application platform for accessing IMS/DB data. All Oracle Connect servers communicate with an Oracle Connect-supplied IMS/TM transaction. This MPP transaction accepts requests from Oracle Connect servers and performs the DLI requests on their behalf.

SQL Support

Oracle Database Gateway for IMS supports SQL-based data access, allowing developers and applications to use this common and standard syntax for retrieving and updating IMS 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 IMS data. This means that Oracle Database Gateway for IMS 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 IMS 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 IMS 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 IMS 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 IMS, 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 IMS offers robust SQL support and relational access to non-relational, proprietary, legacy IMS data. The following topics provide more information about specific SQL features and their support by the gateway.

This section includes the following topics:

- Naming Rules
- SQL Execution

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

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 IMS/DB 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 `IMS`. You must not use an explicit owner qualifier to reference IMS/DB 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 IMS 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.

SQL Execution

Query issues include the following:

- Empty Strings
- Empty Bind Variables

Empty Strings

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

Comparing to an empty string

The gateway passes literal empty string to IMS without any conversion. If you intended an empty string to represent a null value, IMS 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"@IMS where "ename" IS NULL;
```

Selecting an empty string

For VARCHAR columns, the gateway returns an empty string to the Oracle database server 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 IMS as a NULL value.

Handling Non-Relational Data

Non-relational data sources 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.

The metadata is imported and maintained using Oracle Studio for IMS, VSAM, and Adabas Gateways. If COBOL copybooks describing the data source records are available, you can import the metadata by using the metadata import procedure in the Design perspective, on the Metadata tab. If the metadata is provided in a number of COBOL copybooks that use different filter settings, you first import the metadata from copybooks with the same settings and later the metadata from other copybooks.

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.

Hierarchical Modelling

The IMS/DB data sources map the hierarchical model of IMS/DB to the relational model in the following manner:

- Every segment is mapped to a table.
- The fields in a table consist of the IMS segment buffer and the IMS keyfeedback area.
- The index for an IMS 'table' consists of the keyfeedback, i.e. the entire path leading to the specific segment.

The Hospital Database Example includes a simple hierarchy HOSPITAL > WARD > PATIENT. The following figures show the relational model of this three-level hierarchy in Oracle Connect.

Figure 2–1 HOSPITAL in Relational Model

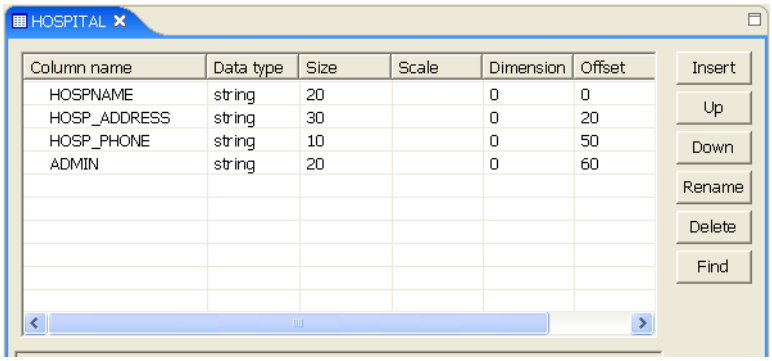


Figure 2–2 WARD in Relational Model

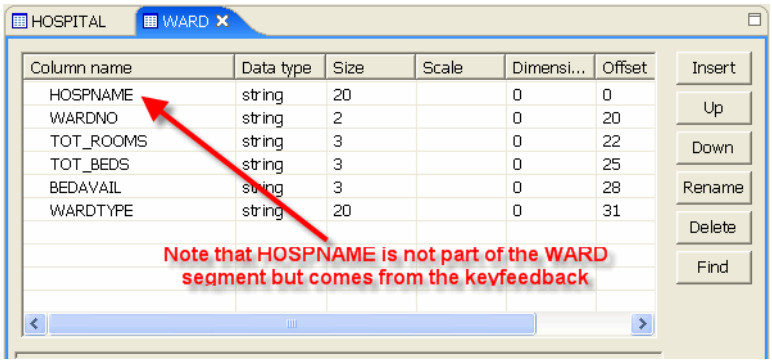
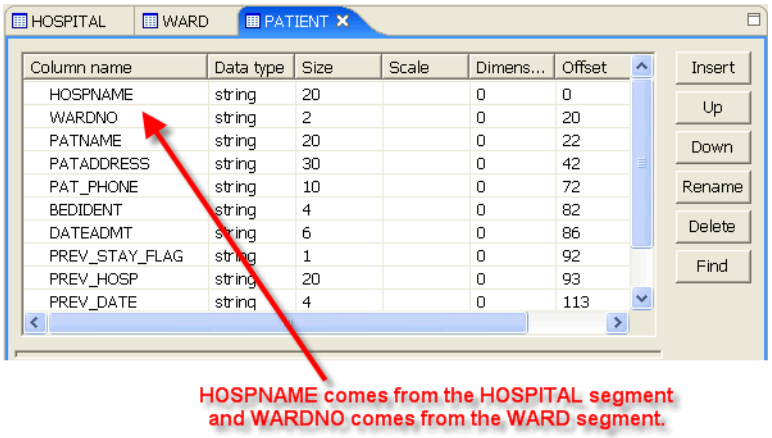


Figure 2–3 PATIENT in Relational Model



Constructing DLI Commands from SQL Requests

When accessing any segment, the data source driver first needs to select a PCB to be used for this purpose. The choice of PCB is made according to the metadata. The metadata import includes the PSB as one of the import sources. As a result, each table definition in the Oracle Connect data dictionary includes a list of PCB numbers that can be used for every table.

For example, in the following figure, PCB0 will be used to access the HOSPITAL database. Note that you can have several PCBs for each table if your PSB includes several PCBs for the same database.

Figure 2–4 PCB Selection

```
<?xml version="1.0" encoding="UTF-8"?>
<table name="HOSPITAL" organization="index">
    <dbCommand>
        <tableSpec dbdName="HOSPDBD" segmentName="HOSPITAL" level="0">
            <pcbList>0</pcbList>
        </tableSpec>
    </dbCommand>
```

In addition, the IMS data source employs a small but effective 'vocabulary' of IMS commands and SSA variations to satisfy incoming requests.

Restrictions

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

- SQL Restrictions
- IMS/DB Restrictions

SQL Restrictions

Restrictions related to SQL are described in the following sections:

- Unsupported SQL Functions
- Transaction Capability
- Transactional Integrity
- Pass-Through Feature
- Table and Column Names
- Database Links
- Stored Procedures

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
CONNECT BY clause	The gateway does not support the `CONNECT BY` clause in a `SELECT` statement.
ROWID	The Oracle ROWID implementation is not supported.

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
```

By default, the IMS-DBCTL data source supports global transactions. If the IMS-DBCTL data source is configured differently in Oracle Studio for IMS, VSAM, and Adabas Gateways, see the *Oracle Database Heterogeneous Connectivity Administrator's Guide* for configuration information.

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

Transactional Integrity

If the IMS data source is defined as auto-commit (the IMS-DLI and IMS-DBDC data sources support only auto-commit), the gateway cannot guarantee transactional integrity. In this case, each UPDATE, INSERT, and DELETE statement is immediately committed on the IMS side and cannot be rolled back.

> **Note:** You can choose to run the gateway for the the IMS-DLI data source in read-only mode by setting the HS_FDS_TRANSACTION_MODE parameter to `READ_ONLY`. Carefully weigh the advantages and disadvantages of executing updates on the IMS data source using the gateway.

Pass-Through Feature

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

Table and Column Names

The gateway metadata defined for IMS allows record and field names to be over 30 characters in length, but the Oracle database server limits both table and column names to 30 characters. Because of this difference, if you access an IMS field with a name defined in the gateway metadata that is over 30 characters long, the name is truncated to 30 characters and the following Oracle error message is returned:

```
ORA-00972: Identifier is too long
```

To avoid this incompatibility, define all names in the gateway metadata less than or equal to 30 characters.

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

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.

Stored Procedures

Stored procedures are not supported.

IMS/DB Restrictions

When accessing IMS/DB data, the following restrictions aply:

- General IMS/DB Restrictions
- Restrictions Specific to IMS-DLI
- Restrictions Specific to IMS-DBCTL
- Restrictions Specific to IMS-DBDC

General IMS/DB Restrictions

The following restrictions apply to all IMS/DB data sources:

- Supported are only databases that have a unique key for every element in a hierarchy, except for end-segments. The support of end-segments without a unique key is limited to read only, with no array (OCCURS clause) support.
- DDL is not supported.
- UPDATE operations on arrays (OCCURS clauses) are not supported. See Normalizing Non-Relational Data on page 3-1 for details on handling arrays.
- Logical databases are supported.
- Logical children are not supported.
- Secondary indexes are not supported.
- Segments whose key is partitioned to several fields in the COBOL layout are supported by exposing both the COBOL-level fields and an additional field that overlays these fields and spans the entire key. The following restrictions apply:
  - Only queries that refer to the overlaid field in the WHERE clause develop an efficient execution strategie.
  - For such a key, only alphanumeric field types are supported.
- CREATE TABLE operations are not supported.
- IMS/TM transactions turn lowercase letters in IMS/DB data into uppercase letters.

Restrictions Specific to IMS-DLI

The following restrictions apply to the IMS-DLI data source (batch access) only:

- The IMS-DLI data source must be set up in a separate workspace for every PSB accessed because the PSB is explicitely coded in the started task JCL.
- Transactional operations, such as COMMIT and ROLLBACK, are not supported. All DML operations are therefore non-transactional.
- UPDATE operations are supported, but it is not recommended to run them over several servers in parallel.

Restrictions Specific to IMS-DBCTL

The following restrictions apply to the IMS-DBCTL data source only:

Segments within a non-unique index are not supported on any level.

Restrictions Specific to IMS-DBDC

The following restrictions apply to the IMS-DBDC data source only:

- Transaction processing is not supported.
- Every data source definition can work with a single PSB only. Multiple data source can be created for multiple PSBs.
- Segments within a non-unique index are not supported at any level.

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 UNIX* 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.

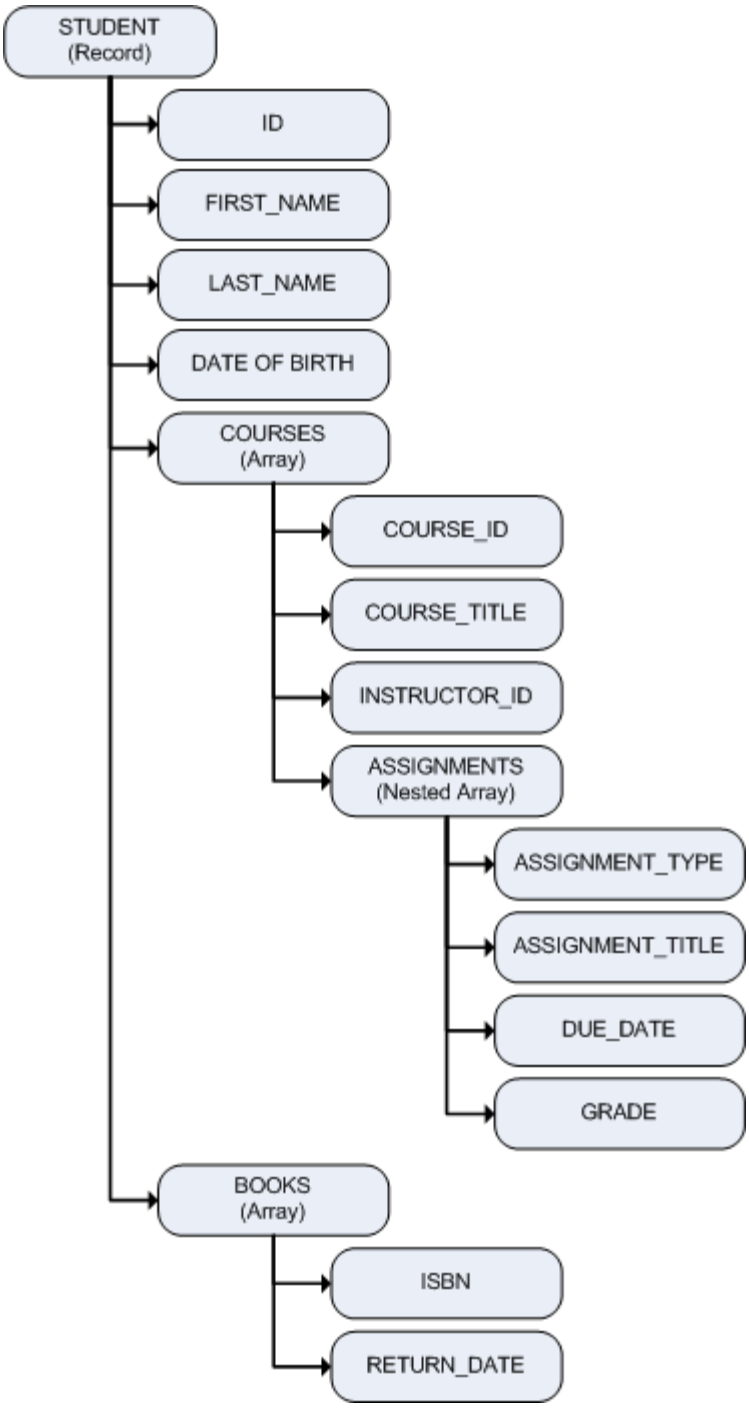
Representing Metadata

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.

See Also: COBOL Copybook Example on page E-1

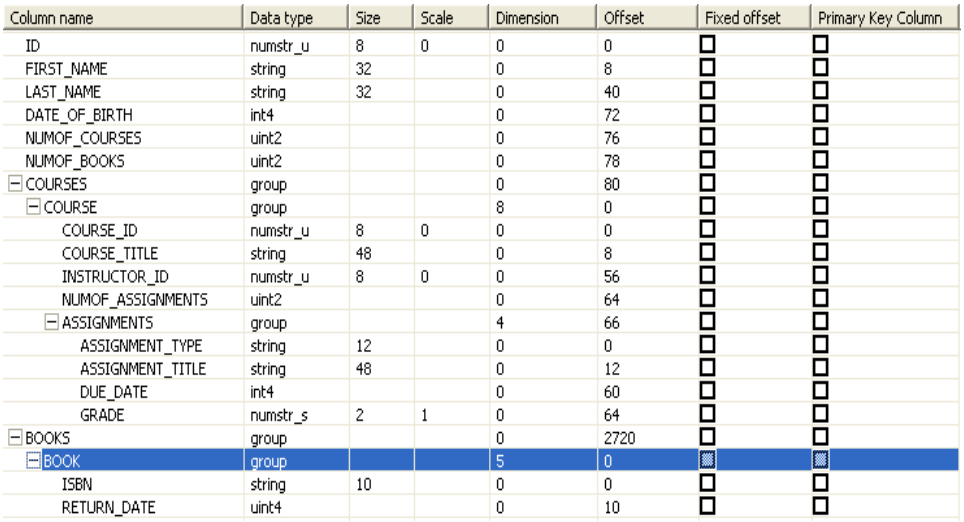
Figure 3–1 Arrays and Nested Arrays



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.

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.

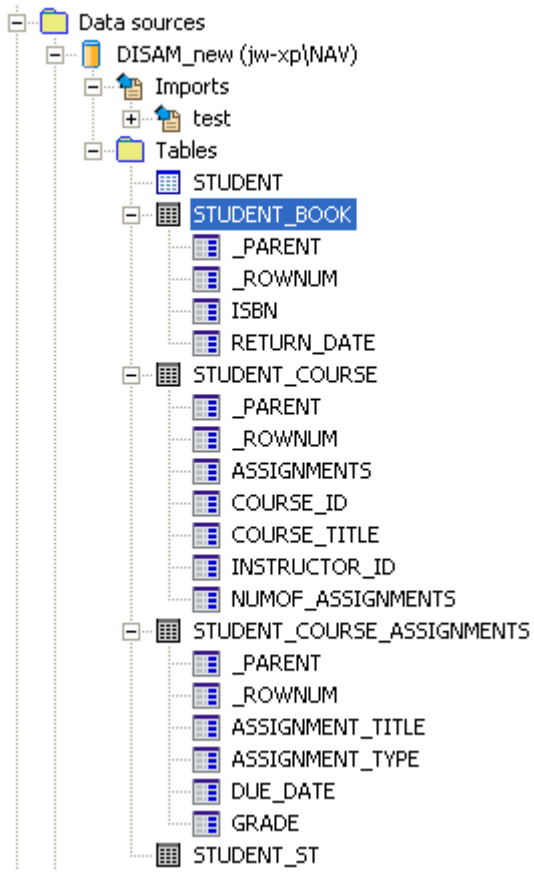
The (+) to the left of a column indicates a group field. This type of field usually has a Dimension value. This value is not mandatory, but it optimizes the access to an array. You can click (+) to display the group members. In Figure 3–2, all groups are expanded.

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.
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.

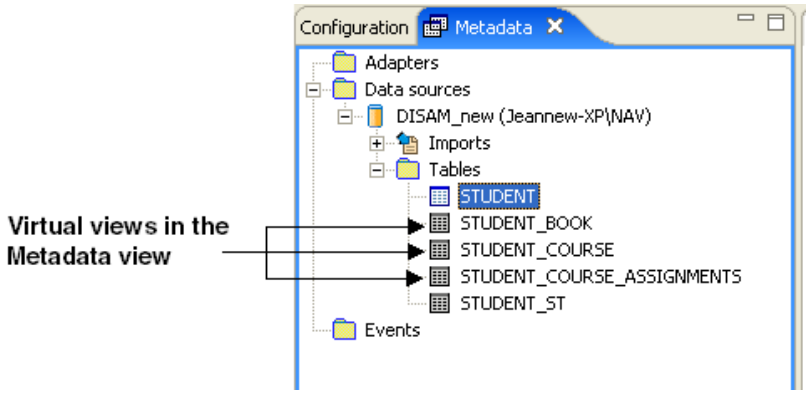
When working with virtual views, consider the following limitations:

- 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 UNIX* 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:

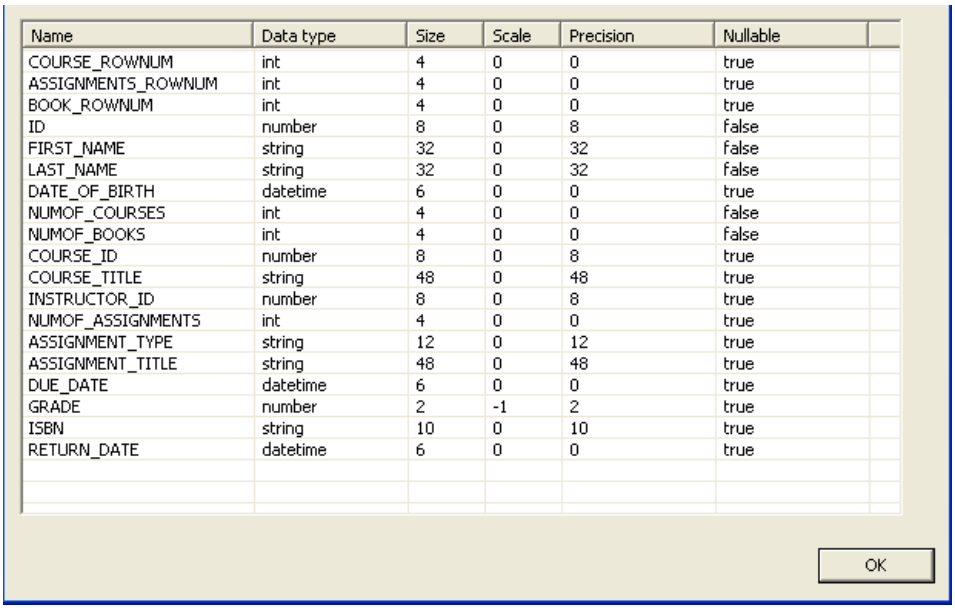
- The parent fields, that is the non-repeating fields.
- The array fields for all arrays within the parent.
- For each array, an optional column called `<array_name>_ROWNUM`, which identifies the row in the array. This column is generated automatically for the array.

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

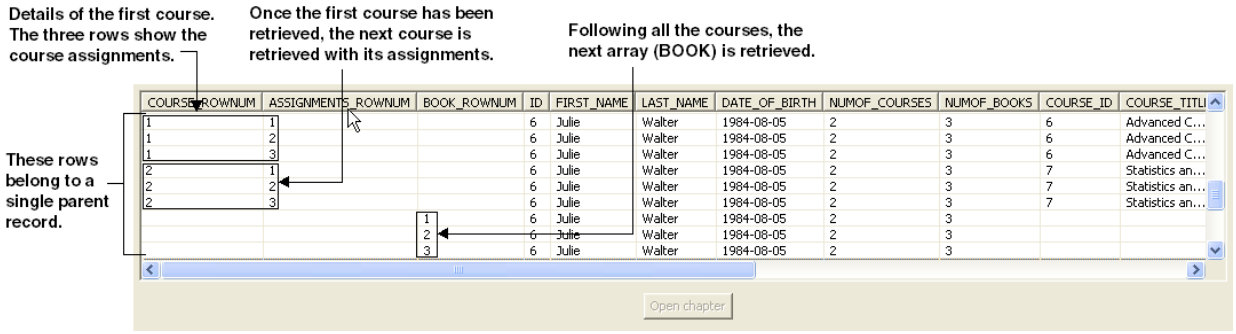
Information Type	Description
Name	The name of the column.
Data Type	The data type supported by that column. For example, string or integer.
Size	Indicates the maximum size allowable for the data in the column. The size is in standard units for the data type. For example, a string with size 40 can have no more than forty characters.
Scale	Indicates the number of digits allowed after the decimal point for a numeric value.

Table 3–2 (Cont.) SQL View Window

Information Type	Description
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



A 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

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 UNIX* for details on configuring data source properties.

Perform the following steps to set the default behavior:

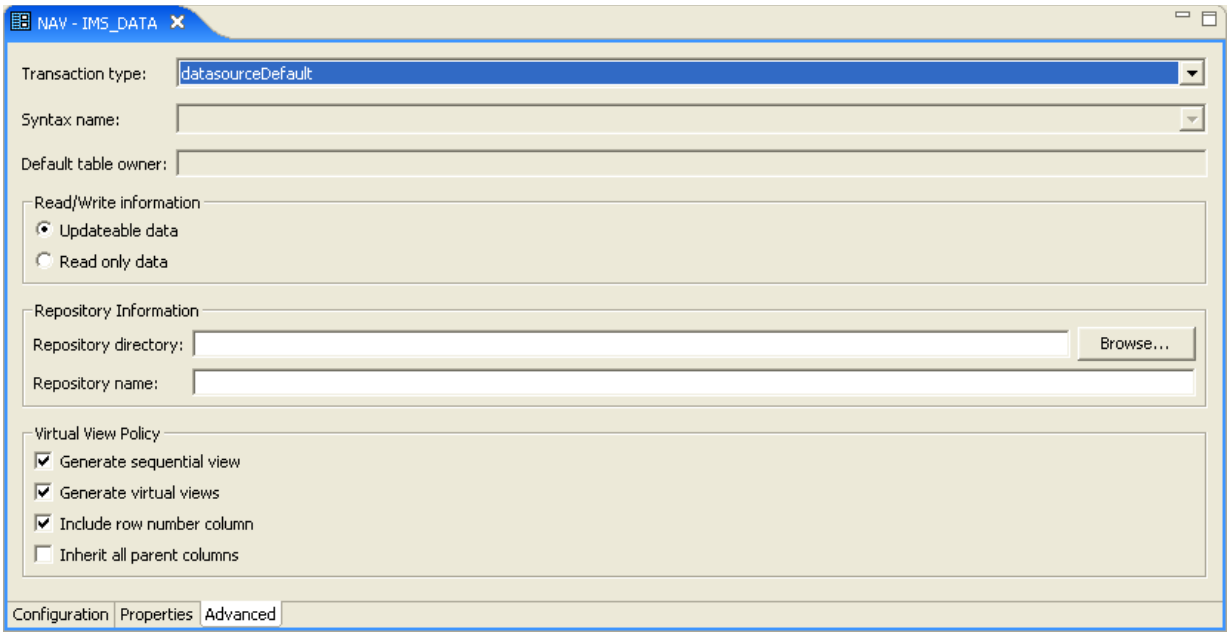
1. Right-click the data source and select **Edit Data Source**.

   The Configuration Properties screen opens.

2. 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



3. Save your settings.

Modifying the Default Behavior During Metadata Import

During the metadata import procedure, the Metadata Model Selection step lets you modify the array handling policy for the tables to be imported. You can define specific settings per table or keep the default values, which are inherited from the settings that you defined on the data source level (see Setting the Default Behavior for Array Handling). After importing a table, you can change its array handling settings on the **Modeling** tab of the table editor (see Modifying the Array Handling Policy for a Specific Table).

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 UNIX* for details on the import procedure per data source.

Perform the following steps to select the metadata model.

1. In the Import Wizard, click **Next** until you reach the Metadata Model Selection step.
2. Select from the following:
   - **Default values for all tables**: Select this option if you want to apply the same values to all tables to be imported.

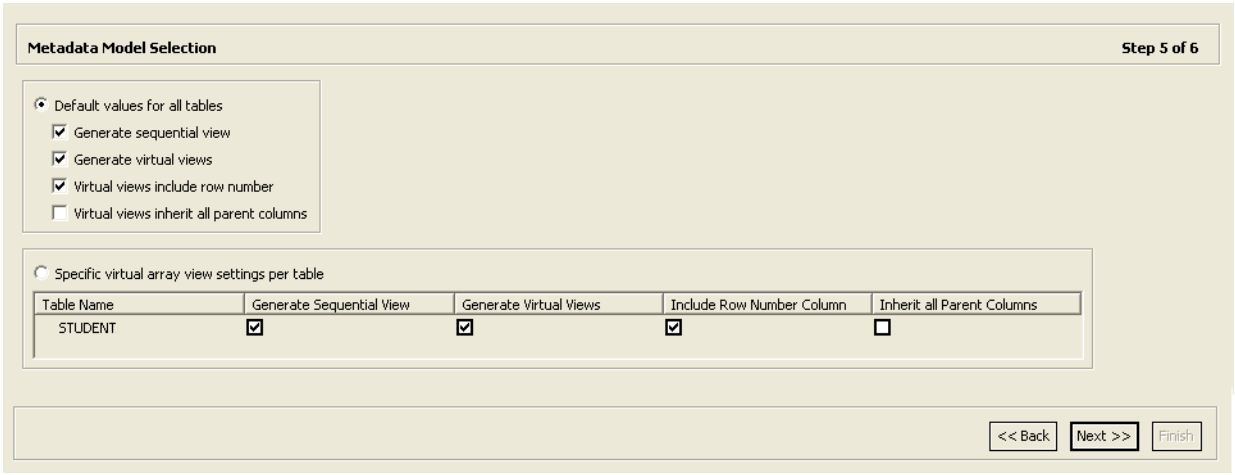
     Note: Selecting this check box discards all table-specific settings.

     – **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.

  - **Virtual views include row number**: Select this option if you want to include a column that specifies the row number in the virtual or sequential view.
  - **Virtual views inherit all parent columns**: Select this option if you want the virtual views to include all the columns of the parent record.
- **Specific virtual array view settings per table**: Select this option if you want to apply different values to the tables to be imported. Then select the relevant check boxes.

Figure 4–2 Selecting the Metadata Model



3. Click **Next** to continue to the next step, then click **Finish**.

Modifying the Array Handling Policy for a Specific Table

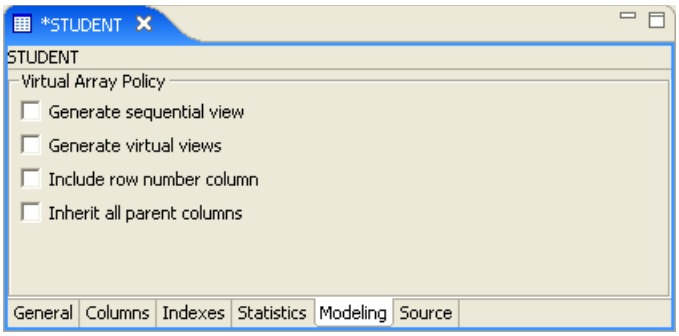
You can also change the array handling policy on the table level. Perform the following steps to modify the policy for a specific table:

1. Right-click the data source and select **Edit metadata**.

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

   The table editor opens.
4. 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 4–3 Modifying the Array Handling Policy on the Table Level



5. 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 Atomic Data Types

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

Table A–1 Atomic Data Types

ADD Data Type	Oracle Data Type	Details
`apt_date`	`DATE`	Date packed into a 4-character string. Format: DMYY Example: 23-Jul-1998 is represented by four bytes: 19, 98, 7, and 23.
`apt_time`	`DATE`	ADD date-time format.
`bit`	`NUMBER(3)`	Single bit within a byte. Size: 1 byte. Format: `datatype="bit"` `onBit="n"` Where *n* specifies which bit (within a byte) the field uses. If more than one bit is defined, then the additional bits may be defined sequentially within the same byte (or bytes, if the number of bits requires this much space).
`bits`	`NUMBER(3)`	A signed number of bits within a byte. Size: 1 bit to 1 byte Format: `<field name="name" datatype="bits" onBit="n" size="m"/>` Where *n* specifies which bit (within a byte) to start from and *m* is the number of bits. If *n* is not specified then n defaults to 1 for the first occurrence of the field and is contiguous thereafter. The maximum number of bits you can map is 32.

Table A–1 (Cont.) Atomic Data Types

ADD Data Type	Oracle Data Type	Details
cstring	VARCHAR2	A null-terminated string of alphanumeric characters; maximum length must be specified. An extra byte is required for the null flag.
cv_datetime	DATE	CorVision date-time format.
date	DATE	ODBC date format.
date6	DATE	Date in a string having the form YYMMDD
date8	DATE	Date in a string having the form YYYYMMDD
dfloat	FLOAT(49)	Double floating-point number (D_FLOAT) Size: 8 bytes Range: 0.29E-38 to 1.7E38 Precision: 16 digits.
filler	RAW	Allocation for future use, string type; length must be specified.
fixed_cstring	FLOAT(49)	A fixed null-terminated string of numeric characters; length must be specified. An extra byte is required for the null flag.
int1	NUMBER(3)	Signed byte integer. Size: 1 byte Range: -128 to +127
int2	NUMBER(5)	Signed word integer. Size: 1 byte Range: -32768 to +32767
int3	NUMBER(10)	Signed integer Size: 3 bytes
int4	NUMBER(10)	Signed long integer. Size: 4 bytes Range: -2147483648 to +2147483647
int6	NUMBER(10)	Signed integer Size: 6 bytes
int_date	DATE	Date in a four-byte integer having the form YYMMDD or YYYYMMDD. Example, 23-Jul-1998 has the form: 980723 or 19980723.
jdate	DATE	Julian date Size: 2 bytes Bits 0-6: (non-century) year Bits 7-15: day of the year
logical	NUMBER(10)	Signed long integer Values: 1 for true, 0 for false
mvs_date	DATE	z/OS date format.
mvs_datetime	DATE	z/OS date-time format.
mvs_time	DATE	z/OS time format.

Table A–1 (Cont.) Atomic Data Types

ADD Data Type	Oracle Data Type	Details
padded_str_date	DATE	Padded date format, not null-terminated.
padded_str_datetime	DATE	Padded date format, not null-terminated.
padded_str_time	DATE	Padded date format, not null-terminated.
phdate	DATE	Size: 2 bytes Bits 0-6: (non-century) year Bits 7-10: number of month Bits 11-15: day of month
single	FLOAT(23)	Single floating-point number (F_FLOAT). Size: 4 bytes Range: 0.29E-38 to 1.7E38 Precision: 6 digits
str_date	CHAR(10)	Atomic date string Size: 10 characters Format: YYYY-MM-DD
str_datetime	CHAR(19)	Atomic date-time string. Size: 19 characters Format: YYYY-MM-DD HH:MM:SS
str_time	CHAR(8)	Atomic date string Size: 8 characters Format: HH:MM:SS
string	CHAR	String of alphanumeric characters; length must be specified.
time	DATE	ODBC time format.
timestamp	DATE	ODBC date-time format.
ubits	NUMBE(3)	An unsigned number of bits within a byte. Size: 1 bit to 1 byte Format: `<field name="name" datetype="bits" onBit="n" size="m"/>` Where *n* specifies which bit (within a byte) to start from and *m* is the number of bits. If *n* is not specified, then *n* defaults to1 for the first occurrence of the field and its contiguous thereafter. The maximum number of bits you can map is 31.
uint1	NUMBER(3)	Unsigned byte integer. Size: 1 byte Range: 0 to +254
uint2	NUMBER(10)	Unsugned word integer. Size: 2 bytes Range: 0 to+65534
uint4	NUMBER(11)	Signed long integer Size: 4 bytes Range: 0 to +4,294,967,294

Table A–1 (Cont.) Atomic Data Types

ADD Data Type	Oracle Data Type	Details
uint6	NUMBER	Unsugned integer. Size: 6 bytes
unicode	VARCHAR2	A null-terminated alphanumeric unicode string. Maximum length must be specified.
varstring	VARCHAR2	16-bit count, followed by a string.
varstring4	VARCHAR2	32-bit count, followed by a string.

Conversion of Non-Atomic Data Types

The gateway converts atomic data types to Oracle data types as described in Table A–1.

Table A–2 Non-atomic Data Types

ADD Data Type	Oracle Data Type	Details
binary	RAW	Unknown data type, string type; length must be specified.
nls_string	CHAR	A string based on language and driven by table.

Conversion of Decimal Data Types

The gateway converts decimal data types to Oracle data types as described in Table A–3.

Table A–3 Decimal Data Types

ADD Data Type	Oracle Data Type	Details
decimal	NUMBER	Packed decimal Maximum number of digits: 31 Maximum fractions: 11 Length = int (number of digits/2) + 1 bytes
numeric_cstring	NUMBER	A null-terminated string of numeric characters; maximum length must be specified. An extra byte is required for the null flag.
numstr_bdn	NUMBER	Signed numeric string. Sign is the first character of the string. Maximum number of digits: 31 Maximum fractions: 11 **Note**: the number of fractions includes the decimal point.
numstr_nlo	NUMBER	Signed numeric string. A left overpunched sign is implemented. Maximum number of digits: 31 Maximum fractions: 11
numstr_nl	NUMBER	Signed numeric string. Sign is the first character of the string. Maximum number of digits: 31 Maximum fractions: 11
numstr_nr	NUMBER	Signed numeric string. Sign is the last character of the string. Maximum number of digits: 31 Maximum fractions: 11

Table A–3 (Cont.) Decimal Data Types

ADD Data Type	Oracle Data Type	Details
numstr_s	NUMBER	Signed numeric string. A right overpunched sign is implemented. Maximum number of digits: 31 Maximum fractions: 11 The number must be right-justified (for example, " 1234N" is -12345). The number can be left-padded by either spaces or zeros. If a scale is provided, it is a fixed positional scale. No decimal point is provided in the data (for example, a value of "1234E" with scale 2 is interpreted as "123.45").
numstr_u	NUMBER	Unsigned numeric string. Maximum number of digits: 31 Maximum fractions: 11
numstr_zoned	NUMBER	Signed numeric string Maximum number of digits: 31 Maximum fractions: 11

Conversion of Scaled Data Types

The gateway converts scaled data types to Oracle data types as described in Table A–4.

Table A–4 Scaled Data Types

ADD Data Type	Oracle Data Type	Details
scaled_int1	NUMBER	Signed byte integer. Size: 1 byte Range: -128 to +127 Maximum: 3
scaled_int2	NUMBER	Singlwe word integer. Size: 2 byts Range: -32768 to +32767 Maximum: 5
scaled_int3	NUMBER	Signed integer. Size: 3 bytes
scaled_int4	NUMBER	Signed long integer. Size: 4 bytes Range: -2147483648 to +2147483647 Maximum: 10
scaled_int6	NUMBER	Signed integer Size: 6 bytes
scaled_int8	NUMBER	Signed quadword. Size: 4 bytes Range: -2147483648 to +2147483647 Maximum: 10

Table A–4 (Cont.) Scaled Data Types

ADD Data Type	Oracle Data Type	Details
`scaled_uint1`	`NUMBER`	Unsigned byte integer. Size: 1 byte Range: 0 to +254 Maximum: 3
`scaled_uint2`	`NUMBER`	Unsigned word integer. Size: 2 bytes Range: 0 to +65534 Maximum: 5
`scaled_uint4`	`NUMBER(5)`	Unsigned long integer Size: 4 bytes Range: 0 to +4,294,967,294 Maximum: 10

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 IMS 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 IMS database.

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

DELETE

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

> **See Also:** "Functions Supported by IMS" on page B-2 for a list of supported functions.

INSERT

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

> **See Also:** "Functions Supported by IMS" on page B-2 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 IMS 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 IMS" on page B-2 for a list of supported functions.

Oracle Functions

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

Functions Not Supported by IMS

Oracle SQL functions with no equivalent function in IMS are not supported in `DELETE`, `INSERT`, or `UPDATE` statements, but are evaluated by the Oracle database server if the statement is a `SELECT` statement. That is, the Oracle database server 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
```

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

Functions Supported by IMS

The gateway translates the following Oracle database server functions in SQL statements to their equivalent IMS functions:

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

Arithmetic Operators

The following table specifies the arithmetic operators.

Oracle	IMS
+	+
-	-

Oracle	IMS
*	*
/	/

Comparison Operators

The following table specifies the comparison operators.

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

Group Functions

The following table specifies the group functions.

Oracle	IMS
AVG	AVG
COUNT	COUNT
MAX	MAX
MIN	MIN
SUM	SUM

String Functions

The following table specifies the string functions.

Oracle	IMS
||	+ (*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	IMS
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 IMS translates a query that refers to an Oracle database server data dictionary table into a query that retrieves the data from an IMS 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 server. The gateway data dictionary is similar to the Oracle database server data dictionary in appearance and use.

This appendix contains the following sections:

Data Dictionary Support

Data dictionary information is stored in the gateway's advanced data dictionary (ADD). 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 paragraphs describe the Oracle Database Gateway for IMS data dictionary support.

Accessing the Gateway Data Dictionary

Accessing an ADD record definition is identical to accessing a data dictionary table or view definition on an Oracle database. You issue a SQL `SELECT` statement specifying a database link. The Oracle database server 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@IMS;
```

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

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 server data dictionary table.
4. Passes the data dictionary information to the Oracle database server.

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_TAB_COMMENTS
- USER_OBJECTS
- 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 server data dictionary tables or views are supported. If you use a view not on the list, you receive the Oracle database server error code for no more rows available.

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 IMS even when access privileges for those IMS 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 server data dictionary. The Oracle database server columns that are missing in an IMS data dictionary table are filled with zeros, spaces, null values, or default values, depending on the column type.

IMS 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 server 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 IMS.

Table C–1 describes the ALL_CATALOG table.

Table C–1 ALL_CATALOG

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	IMS
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)	IMS

Table C–2 (Cont.) ALL_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–3 describes the ALL_CONS_COLUMNS table.

Table C–3 ALL_CONS_COLUMNS

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2(30)	-
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	VARCHAR(3)	IMS
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	-	VARCHAR(3)	-
R_CONSTRAINT_NAME	-	VARCHAR2(30)	-
DELETE_RULE	NOT NULL	VARCHAR(7)	-
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)	-

Table C–5 (Cont.) ALL_IND_COLUMNS

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

Table C–6 (Cont.) ALL_INDEXES

Name	Null?	Type	Value
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)	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)	IMS
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

Table C–7 (Cont.) ALL_OBJECTS

Name	Null?	Type	Value
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–8 describes the ALL_TAB_COLUMNS table.

Table C–8 ALL_TAB_COLUMNS

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

Table C–10 (Cont.) ALL_TABLES

Name	Null?	Type	Value
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–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)	IMS
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)	-

Table C–12 (Cont.) ALL_VIEWS

Name	Null?	Type	Value
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)	IMS
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)	IMS
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)	IMS
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)	-

Table C–15 (Cont.) DBA_TAB_COLUMNS

Name	Null?	Type	Value
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–16 describes the DICT_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)	IMS
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)	IMS
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)	-

Table C–22 (Cont.) USER_CONSTRAINTS

Name	Null?	Type	Value
BAD	NOT NULL	CHAR(1)	-
RELY	NOT NULL	CHAR(1)	-
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

Table C–24 (Cont.) USER_INDEXES

Name	Null?	Type	Value
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)	-
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

Table C–25 (Cont.) USER_OBJECTS

Name	Null?	Type	Value
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–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)	-

Table C–26 (Cont.) USER_TAB_COLUMNS

Name	Null?	Type	Value
USER_STATS	NOT NULL	CHAR(1)	-
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

Table C–28 (Cont.) USER_TABLES

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

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 UNIX*.

To set the language in Studio

1. In the Design perspective, open the machine for which you want to set the language.
2. Open the Bindings list 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	
AR8EBCDICX	Arabic + Latin	AR8ISO8859P6	

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

EBCDIC CP Name	Description	Base ASCII CP	Multibyte
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	
WE8EBCDIC924	Latin 9	WE8ISO8859P9	

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

EBCDIC CP Name	Description	Base ASCII CP	Multibyte
ZHS16DBCS	Simplified Chinese	ZHS16CGB231280	Yes
ZHT16DBCS	Traditional Chinese	ZHT16BIG5	Yes

E

Examples

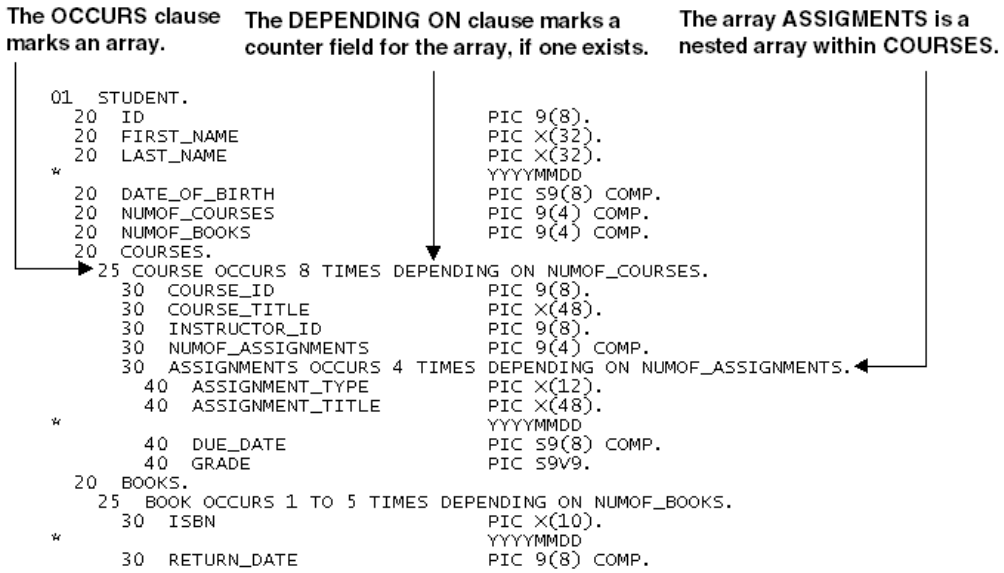
This section contains the following topics:

- COBOL Copybook Example
- Hospital Database Example

COBOL Copybook Example

Figure E–1 shows a COBOL copybook that illustrates arrays and nested arrays.

Figure E–1 Metadata Example in COBOL



Hospital Database Example[1]

The following are the full source files for the Hospital database example that is used in Hierarchical Modelling on page 2-4 and Constructing DLI Commands from SQL Requests on page 2-5.

[1] Kapp, Dan and Leben, Joe: **IMS Programming Techniques**. Van Nostrand Reinhold Company Inc., New York, 1986.

Example E–1 Hospital Cobol Copybook

```
01  HOSPITAL.
  03  HOSPNAME        PIC X(20).
  03  HOSP-ADDRESS      PIC X(30).
  03  HOSP-PHONE        PIC X(10).
  03  ADMIN         PIC X(20).

01  WARD.
  03  WARDNO          PIC XX.
  03  TOT-ROOMS       PIC XXX.
  03  TOT-BEDS        PIC XXX.
  03  BEDAVAIL        PIC XXX.
  03  WARDTYPE        PIC X(20).

01  PATIENT.
  03  PATNAME         PIC X(20).
  03  PATADDRESS        PIC X(30).
  03  PAT-PHONE       PIC X(10).
  03  BEDIDENT        PIC X(4).
  03  DATEADMT        PIC X(6).
  03  PREV-STAY-FLAG      PIC X.
  03  PREV-HOSP       PIC X(20).
  03  PREV-DATE       PIC X(4).
  03  PREV-REASON       PIC X(30).

01  SYMPTOM.
  03  DIAGNOSE        PIC X(20).
  03  SYMPDATE        PIC X(6).
  03  PREV-TREAT-FLAG     PIC X.
  03  TREAT-DESC        PIC X(20).
  03  SYMP-DOCTOR       PIC X(20).
  03  SYMP-DOCT-PHONE     PIC X(10).

01  TREATMNT.
  03  TRTYPE          PIC X(20).
  03  TRDATE          PIC X(6).
  03  MEDICATION-TYPE     PIC X(20).
  03  DIET-COMMENT      PIC X(30).
  03  SURGERY-FLAG      PIC X.
  03  SURGERY-DATE      PIC X(6).
  03  SURGERY-COMMENT     PIC X(30).

01  DOCTOR.
  03  DOCTNAME        PIC X(20).
  03  DOCT-ADDRESS      PIC X(30).
  03  DOCT-PHONE        PIC X(10).
  03  SPECIALT        PIC X(20).

01  FACILITY.
  03  FACTYPE         PIC X(20).
  03  TOT-FACIL       PIC XXX.
  03  FACAVAIL        PIC XXX.
```

Example E–2 Hospital DBD

```
  PRINT NOGEN
     DBD NAME=HOSPDBD,ACCESS=HDAM,RMNAME=(DFSHDC40,40,100)
     DATASET DD1=PRIME,DEVICE=3390

     SEGM NAME=HOSPITAL,PARENT=0,BYTES=80
```

```
      FIELD NAME=(HOSPNAME,SEQ,U),BYTES=20,START=1,TYPE=C
      FIELD NAME=ADMIN,BYTES=20,START=61,TYPE=C

    SEGM NAME=WARD,PARENT=HOSPITAL,BYTES=31
      FIELD NAME=(WARDNO,SEQ,U),BYTES=2,START=1,TYPE=C
      FIELD NAME=BEDAVAIL,BYTES=3,START=9,TYPE=C
      FIELD NAME=WARDTYPE,BYTES=20,START=12,TYPE=C

    SEGM NAME=PATIENT,PARENT=WARD,BYTES=125
      FIELD NAME=(BEDIDENT,SEQ,U),BYTES=4,START=61,TYPE=C
      FIELD NAME=PATNAME,BYTES=20,START=1,TYPE=C
      FIELD NAME=DATEADMT,BYTES=6,START=65,TYPE=C

    SEGM NAME=SYMPTOM,PARENT=PATIENT,BYTES=77
      FIELD NAME=(SYMPDATE,SEQ),BYTES=6,START=21,TYPE=C
      FIELD NAME=DIAGNOSE,BYTES=20,START=1,TYPE=C

    SEGM NAME=TREATMNT,PARENT=PATIENT,BYTES=113
      FIELD NAME=(TRDATE,SEQ),BYTES=6,START=21,TYPE=C
      FIELD NAME=TRTYPE,BYTES=20,START=1,TYPE=C

    SEGM NAME=DOCTOR,PARENT=PATIENT,BYTES=80
      FIELD NAME=DOCTNAME,BYTES=20,START=1,TYPE=C
      FIELD NAME=SPECIALT,BYTES=20,START=61,TYPE=C

    SEGM NAME=FACILITY,PARENT=HOSPITAL,BYTES=26
      FIELD NAME=FACTYPE,BYTES=20,START=1,TYPE=C
      FIELD NAME=FACAVAIL,BYTES=3,START=24,TYPE=C

    DBDGEN
    FINISH
    END
```

Example E–3 Hospital PSB

```
  PRINT NOGEN
  PCB TYPE=DB,DBDNAME=HOSPDBD,PROCOPT=AP,KEYLEN=32
*
  SENSEG NAME=HOSPITAL,PARENT=0
  SENSEG NAME=WARD,PARENT=HOSPITAL
  SENSEG NAME=PATIENT,PARENT=WARD
  SENSEG NAME=SYMPTOM,PARENT=PATIENT
  SENSEG NAME=TREATMNT,PARENT=PATIENT
  SENSEG NAME=DOCTOR,PARENT=PATIENT
  SENSEG NAME=FACILITY,PARENT=HOSPITAL
*
  PSBGEN LANG=COBOL,PSBNAME=HOSPPSB
  END
```

Index

N

O

P

R

S

T

U