Oracle® Warehouse Builder

API and Scripting Reference

11*g* Release 2 (11.2)

E10584-02

July 2013

ORACLE®

Oracle Warehouse Builder API and Scripting Reference, 11*g* Release 2 (11.2)

E10584-02

Primary Author: Reema Khosla

Contributor: Vishwanath Sreeraman, Antonio Romero, John Leigh, Linda Bittarelli, Frank Yang, Robert Velisar, Lyudmila Mogilevich, Xiaoge Zhang, Gary Tripp, Justin Ho

Contents

7 Creating and Managing Custom Objects and Properties

A Additional and Optional Usages

Preface

This preface includes the following topics:

Purpose

Oracle Warehouse Builder is a comprehensive toolset for practitioners who move and transform data, develop and implement business intelligence systems, perform metadata management, or create and manage Oracle databases and metadata. This guide describes how to use Warehouse Builder Oracle MetaBase (OMB) Scripting Language to:

- Create a definition of a data warehouse.
- Configure the definitions for a physical instance of the data warehouse.
- Validate the set of definitions and their configurations.
- Generate a set of scripts to create and populate the data warehouse instance.
- Generate data transformation scripts.
- Deploy and initially load the data warehouse instance.
- Maintain the physical instance by conditionally refreshing it with generated scripts.
- Integrate Warehouse Builder metadata with other Business Intelligence products.
- Populate Oracle Discoverer EULs and OLAP catalogs for analyzing the data warehouse.

The guide also describes how to create and use Experts. It also lists the run-time and design time public views available in Warehouse Builder. It also describes how users can add new objects called user defined objects to Warehouse Builder.

Audience

This guide is intended for data warehouse practitioners who want to access Warehouse Builder functionality programmatically:

- Business Intelligence application developers

- Warehouse architects, designers, and developers—especially SQL and PL/SQL developers
- Developers of large-scale products based on data warehouses
- Warehouse administrators
- System administrators
- Other MIS professionals

In order to use the information in this guide, you need to be comfortable with the concepts of Relational Database Management Systems and Data Warehouse design. For information on data warehousing, refer to the *Oracle Database Data Warehousing Guide.* Also, you need to be familiar with Oracle's relational database software products such as Oracle Database, SQL*Plus, SQL*Loader, Oracle Enterprise Manager, and Oracle Workflow.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at
http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

The Warehouse Builder documentation set includes these manuals:

- *Oracle Warehouse Builder Concepts*
- *Oracle Warehouse Builder Installation and Administration Guide for Windows and Linux*
- *Oracle Warehouse Builder Sources and Targets Guide*
- *Oracle Warehouse Builder Data Modeling, ETL, and Data Quality Guide*

In addition to the Warehouse Builder documentation, you can refer to other documents listed below:

- *Oracle Database Data Warehousing Guide*

Oracle provides additional information sources, including other documentation, training, and support services that can enhance your understanding and knowledge of Oracle Warehouse Builder.

- For more information on Oracle Warehouse Builder technical support, contact Oracle World Wide Support services at:

  http://www.oracle.com/support

- For the latest information on, and downloads of, software and documentation updates to Oracle Warehouse Builder, visit My Oracle Support at:

  http://support.oracle.com

- You can order other Oracle documentation at:

  http://shop.oracle.com

Conventions

In this manual, Windows refers to the Windows NT, Windows 2000, and Windows XP operating systems. The SQL*Plus interface to Oracle Database may be referred to as SQL.

In the examples, an implied carriage return occurs after each line, unless otherwise noted. You must press the Return key after each line of input.

The following conventions are also used in this manual:

Convention	Meaning
. . .	Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.
...	Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted.
boldface text	Boldface type in text refers to interface buttons and links. Boldface type also serves as emphasis to set apart main ideas.
italicized text	Italicized text applies to new terms introduced for the first time. Italicized text also serves as an emphasis on key concepts.
`unicode text`	Unicode text denotes exact code, file directories and names, and literal commands.
`italicized unicode text`	Italicized unicode text refers to parameters whose value is specified by the user.
[]	Brackets enclose optional clauses from which you can choose one or none.

Part I

Public Views

This part contains the following chapters:

1

Public Views for the Runtime Environment

The Warehouse Builder provides a set of pre-built views for both the design and runtime environments. These views are called the Warehouse Builder public views and are the API alternative to using the Repository Browser described in the *Oracle Warehouse Builder Data Modeling, ETL, and Data Quality Guide*.

Use these views to access to metadata and data stored in Warehouse Builder repositories.

You can access public views by logging in to SQL*Plus with default `workspace@set`. When you login to SQL*Plus, you can access public views (design-time public views or runtime public views) from your default workspace.

If you try to access public views from any workspace other than the default, then you must call: `WB_workspace_management.set_workspace`(<wksp_name>, <wksp_owner>). If you want to switch to a workspace other than the default one, then you can call the `WB_workspace_management.set_workspace` procedure. You must have the `ACCESS_PUBLICVIEW_BROWSER` system privilege to retrieve useful information from the public views. Otherwise, you will get "0 rows returned." You may need to ask the workspace owner or workspace admin to grant the system privilege `ACCESS_PUBLICVIEW_BROWSER`.

This chapter contains a catalog of the public views for the runtime environment.

1.1 Warehouse Builder Runtime Repository Public Views

The Runtime Repository contains all of the deployment and execution audit data. Use these public views to access this data. These views are used by Runtime Audit Browser to provide audit reporting.

Deployment Auditing Views

1.2 Deployment Auditing Views

Table 1–1 ALL_RT_AUDIT_LOCATIONS

Column Name	Data Type	Description
LOCATION_AUDIT_ID	NUMBER (22)	Internal primary key to audit_location
RUNTIME_VERSION	VARCHAR2 (64)	Runtime version number
CLIENT_VERSION	VARCHAR2 (64)	Design client version number
CLIENT_REPOSITORY	VARCHAR2 (30)	Name of the client repository
CLIENT_REPOSITORY_VERSION	VARCHAR2 (64)	Client repository version number
REPOSITORY_USER	VARCHAR2 (30)	Username of the design repository
GENERATION_TIME	DATE	When the deployment was generated
DEPLOYMENT_AUDIT_ID	NUMBER (22)	Internal audit ID of the deployment
DEPLOYMENT_SEQUENCE_NUMBER	NUMBER (10)	Sequence number of this location in the deployment
DEPLOYMENT_AUDIT_NAME	VARCHAR2 (64)	Audit name of the location
DEPLOYMENT_AUDIT_STATUS	VARCHAR2 (4000)	INACTIVE, READY, or COMPLETE

Table 1–1 (Cont.) ALL_RT_AUDIT_LOCATIONS

Column Name	Data Type	Description
LOCATION_AUDIT_STATUS	VARCHAR2 (4000)	INACTIVE, READY, BUSY_PREPARE, BUSY_UNPREPARE, BUSY_DEPLOY, BUSY_UNDO, BUSY_FINALIZE, or COMPLETE
LOCATION_UOID	VARCHAR2 (32)	Client UOID of the location
LOCATION_NAME	VARCHAR2 (64)	Name of the location
LOCATION_TYPE	VARCHAR2 (64)	Type of the location. (ODB, OWF, OEM)
LOCATION_TYPE_VERSION	VARCHAR2 (64)	Version of the target
NUMBER_SCRIPT_RUN_ERRORS	NUMBER (10)	Number of errors detected
NUMBER_SCRIPT_RUN_WARNINGS	NUMBER (10)	Number of warnings detected
CREATED_ON	DATE	The time audit data was created
CREATED_BY	VARCHAR2 (30)	Database username
UPDATED_ON	DATE	The time audit data was updated
UPDATED_BY	VARCHAR2 (30)	Database username

Table 1–2 ALL_RT_AUDIT_LOCATION_MESSAGES

Column Name	Data Type	Description
MESSAGE_AUDIT_ID	NUMBER (22)	Internal key to audit_location_message. Primary when used with message_line_number.
LOCATION_AUDIT_ID	NUMBER (22)	Internal key to audit_location
MESSAGE_SEVERITY	VARCHAR2 (4000)	INFORMATIONAL, WARNING, ERROR, or RECOVERY
MESSAGE_LINE_NUMBER	NUMBER (10)	1 for single line messages >0 for multiple line messages (Forms primary key when used with message_audit_id)
MESSAGE_TEXT	VARCHAR2 (4000)	plain_text or nls_key
CREATED_ON	DATE	The time audit data was created
CREATED_BY	VARCHAR2 (30)	Database username

Table 1–3 ALL_RT_AUDIT_LOCATION_FILES

Column Name	Data Type	Description
FILE_AUDIT_ID	NUMBER (22)	Internal primary key to audit_location_file
LOCATION_AUDIT_ID	NUMBER (22)	Internal key to audit_location
FILE_TYPE	VARCHAR2 (64)	SQLLoaderLogFile, ShellOutputStream, ShellErrorStream, FTPOutputStream, or FTPErrorStream
FILE_TEXT	CLOB	Contents of the file
FORMAT	VARCHAR2(4000)	TEXT or HTML
CREATED_ON	DATE	The time audit data was created
CREATED_BY	VARCHAR2 (30)	Database username

Table 1–4 ALL_RT_AUDIT_OBJECTS

Column Name	Data Type	Description
OBJECT_AUDIT_ID	NUMBER (22)	Internal primary key to audit_object
PARENT_OBJECT_AUDIT_ID	NUMBER (22)	Internal key to parent audit_script_run
LOCATION_AUDIT_ID	NUMBER (22)	Internal key to audit_location
LOCATION_SEQUENCE_NUMBER	NUMBER (10)	Sequence number of this object in the location
OBJECT_UOID	VARCHAR2 (32)	UOID of the deployed object
OBJECT_NAME	VARCHAR2 (64)	Name of the deployed object
OBJECT_TYPE	VARCHAR2 (64)	Type of deployed object. (PLSQLMap, Table, Dimension, SQLLoaderControlFile)
CLIENT_VERSION_TAG	VARCHAR2 (80)	Client version identifier of this object
NUMBER_SCRIPT_RUN_ERRORS	NUMBER (10)	Number of errors detected
NUMBER_SCRIPT_RUN_WARNINGS	NUMBER (10)	Number of warnings detected
STATUS_WHEN_DEPLOYED	VARCHAR2 (4000)	VALID, INVALID, REMOVED, or UNCERTAIN
CREATED_ON	DATE	The time audit data was created
CREATED_BY	VARCHAR2 (30)	Database username
UPDATE_ON	DATE	The time audit data was updated
UPDATED_BY	VARCHAR2 (30)	Database username

Table 1–5 ALL_RT_AUDIT_SCRIPT_MESSAGES

Column Name	Data Type	Description
MESSAGE_AUDIT_ID	NUMBER (22)	Internal primary key to audit_script_file
SCRIPT_RUN_AUDIT_ID	NUMBER (22)	Internal key to audit_script_run
MESSAGE_SEVERITY	VARCHAR2 (4000)	INFORMATIONAL, WARNING, ERROR, or RECOVERY
MESSAGE_LINE_NUMBER	NUMBER (10)	1 for single line messages >0 for multiple line messages (Forms primary key when used with message_audit_id)
MESSAGE_TEXT	VARCHAR2 (4000)	plain_text or nls_key
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username

Table 1–6 ALL_RT_AUDIT_SCRIPT_RUNS

Column Name	Data Type	Description
SCRIPT_RUN_AUDIT_ID	NUMBER (22)	Internal primary key to audit_script_run
LOCATION_AUDIT_ID	NUMBER (22)	Internal key to audit_location
OBJECT_AUDIT_ID	NUMBER (22)	Internal key to audit_object
SCRIPT_RUN_AUDIT_STATUS	VARCHAR2 (4000)	BUSY, COMPLETE, UNCERTAIN, FAILED or INACTIVE
OPERATION	VARCHAR2 (4000)	DEPLOY, or UNDO

Table 1–6 (Cont.) ALL_RT_AUDIT_SCRIPT_RUNS

Column Name	Data Type	Description
SCRIPT_ACTION	VARCHAR2 (4000)	CREATE, DROP, UPGRADE, or REPORT
SCRIPT	CLOB	Script used to perform the action
SCRIPT_FORMAT	VARCHAR2 (4000)	TEXT or HTML
SCRIPT_GENERATION_TIME	DATE	The time the script was created
NUMBER_SCRIPT_RUN_ERRORS	NUMBER	The number of errors detected
NUMBER_SCRIPT_RUN_WARNINGS	NUMBER	The number of warnings detected
ELAPSE_TIME	NUMBER (10)	The number of seconds that elapsed
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username
UPDATED_ON	DATE	The time the audit data updated
UPDATED_BY	VARCHAR2 (30)	Database username

Table 1–7 ALL_RT_AUDIT_SCRIPT_FILES

Column Name	Data Type	Description
FILE_AUDIT_ID	NUMBER (22)	Internal primary key to audit_script_file
SCRIPT_RUN_AUDIT_ID	NUMBER (22)	Internal key to audit_script_run
FILE_TYPE	VARCHAR2 (64)	SQLLoaderLogFile, ShellOutputStream, ShellErrorStream, FTPOutputStream, or FTPErrorStream
FILE_TEXT	CLOB	Contents of the file
FORMAT	VARCHAR2 (4000)	TEXT or HTML
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username

Table 1–8 ALL_RT_AUDIT_DEPLOYMENTS

Column Name	Data Type	Description
DEPLOYMENT_AUDIT_ID	NUMBER(22)	ID of the deployment audit
DEPLOYMENT_AUDIT_NAME	VARCHAR2(64)	Name of the deployment audit
NUMBER_OF_UNITS	NUMBER(10)	Number of units
RUNTIME_VERSION	VARCHAR2(64)	Version of the control center
CLIENT_VERSION	VARCHAR2(64)	Version of the Warehouse Builder client
CLIENT_REPOSITORY	VARCHAR2(30)	Name of client repository
CLIENT_REPOSITORY_VERSION	VARCHAR2(64)	Version of the client repository
REPOSITORY_USER	VARCHAR2(30)	Name of the repository user
GENERATION_TIME	DATE	Timestamp of the object generation
DEPLOYMENT_AUDIT_STATUS	VARCHAR2(4000)	Status of the deployment
NUMBER_SCRIPT_RUN_ERRORS	NUMBER(10)	Number of errors running the scripts within the deployment

Table 1–8 (Cont.) ALL_RT_AUDIT_DEPLOYMENTS

Column Name	Data Type	Description
NUMBER_SCRIPT_RUN_WARNINGS	NUMBER(10)	Number of warnings while running the scripts within the deployment
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 1–9 ALL_RT_INSTALLATIONS

Column Name	Data Type	Description
INSTALLATION_ID	VARCHAR2	ID of the installation
INSTALLATION_NAME	CHAR	Name of the installation
DESCRIPTION	CHAR	Description of the installation
INSTALLED_VERSION	VARCHAR2(4000)	Version of the repository that is installed
RELEASE	VARCHAR2	Release number of Warehouse Builder
PUBLIC_VIEW_VERSION	CHAR(4)	Version of the PublicViews implemented by this installation
SCHEMA_TYPE	VARCHAR2(24)	Type of schema
UPDATED_ON	VARCHAR2	Update timestamp
CREATED_ON	VARCHAR2	Creation timestamp

Table 1–10 ALL_RT_LOCATIONS

Column Name	Data Type	Description
LOCATION_NAME	VARCHAR2(64)	Name of the location
LOCATION_UOID	VARCHAR2(32)	UOID of the location
IS_DEPLOYMENT_TARGET	VARCHAR2(5)	Indicates whether this location is a deployment target
LOCATION_TYPE	VARCHAR2(64)	Type of the source or target with which the location is associated
LOCATION_TYPE_VERSION	VARCHAR2(64)	Version of the source or target

Table 1–11 ALL_RT_LOCATION_PARAMETERS

Column Name	Data Type	Description
LOCATION_NAME	VARCHAR2(64)	Name of the location
PARAMETER_NAME	VARCHAR2(64)	Name of the parameter
IS_ACCESS_RESTRICTED	VARCHAR2(5)	Indicates whether the parameter-value has a restricted value, such as for a password.
PARAMETER_VALUE	VARCHAR2(4000)	The value of a non-access restricted parameter

Table 1–12 ALL_RT_OBJECTS

Column Name	Data Type	Description
OBJECT_NAME	VARCHAR2(64)	Name of the object
OBJECT_TYPE	VARCHAR2(64)	Type of the object
PARENT_OBJECT_NAME	VARCHAR2(64)	Name of the parent object
PARENT_OBJECT_TYPE	VARCHAR2(64)	Type of the parent object
LOCATION_NAME	VARCHAR2(64)	Name of the location to which this object is deployed
OBJECT_UOID	VARCHAR2(32)	UOID of the object
VERSION_TAG	VARCHAR2(80)	Version string used by Control Center Manager
LAST_DEPLOYMENT_SCRIPT	CLOB	Script that was used during this objects last deployment
STATUS_WHEN_LAST_ DEPLOYED	VARCHAR2(4000)	Object status after its last deployment
SUB_OBJECTS_VALID	VARCHAR2(5)	Indicates whether the subobjects of an object were valid at last deployment
RELATED_OBJECTS_VALID	VARCHAR2(5)	Indicates whether the related objects of an object were valid at last deployment
DEPLOYMENT_DATE	DATE	Date the object was deployed
DEPLOYED_BY	VARCHAR2(30)	User who deployed the object

Table 1–13 ALL_RT_TASKS

Column Name	Data Type	Description
CONTEXT_OBJECT_NAME	VARCHAR2(64)	Name of the tasks context
CONTEXT_OBJECT_TYPE	VARCHAR2(64)	Type of the tasks context
CONTEXT_LOCATION_NAME	VARCHAR2(64)	Name of the location where the context was deployed to
EXEC_LOCATION_NAME	VARCHAR2(64)	Name of the location where the task will be executed
TASK_NAME	VARCHAR2(64)	Name of the task
TASK_TYPE	VARCHAR2(64)	Type of the task
TASK_INPUT	CLOB	Script that implements the task
OBJECT_NAME	VARCHAR2(64)	Name of the object
OBJECT_TYPE	VARCHAR2(64)	Type of the object
OBJECT_LOCATION_NAME	VARCHAR2(64)	Location to which the object is deployed

Table 1–14 ALL_RT_TASK_PARAMETERS

Column Name	Data Type	Description
CONTEXT_OBJECT_NAME	VARCHAR2(64)	Name of the tasks context
CONTEXT_OBJECT_TYPE	VARCHAR2(64)	Type of the tasks context
CONTEXT_LOCATION_NAME	VARCHAR2(64)	Name of the location where the context was deployed to
EXEC_LOCATION_NAME	VARCHAR2(64)	Name of the location where the task will be executed

Table 1–14 (Cont.) ALL_RT_TASK_PARAMETERS

Column Name	Data Type	Description
TASK_NAME	VARCHAR2(64)	Name of the task
PARAMETER_KIND	VARCHAR2(4000)	Kind of Parameter
CUSTOM_PARAMETER_UOID	VARCHAR2(32)	UOID of the parameter if it is a custom parameter
PARAMETER_NAME	VARCHAR2(64)	Name of the parameter
PARAMETER_TYPE	VARCHAR2(4000)	Type of the parameter
PARAMETER_MODE	VARCHAR2(4000)	Mode of the parameter (IN/OUT/INOUT/VARIABLE)
PARAMETER_SCOPE	VARCHAR2(4000)	Scope of the parameter (GLOBAL, SHARED, PARAMETER, LOCAL or INNER)
BOUND_TO_NAME	VARCHAR2(64)	Name of object that the parameter might be bound to
BOUND_TO_KIND	VARCHAR2(4000)	Kind of object that the parameter might be bound to
BOUND_TO_SCOPE	VARCHAR2(4000)	Scope of the object that the parameter might be bound to
IS_FIXED	VARCHAR2(5)	Is parameter value fixed
IS_REQUIRED	VARCHAR2(5)	Is parameter value required
TYPE_LENGTH	NUMBER(10)	Data type length of parameter
TYPE_SCALE	NUMBER(10)	Data type scale of parameter
TYPE_PRECISION	NUMBER(10)	Data type precision of parameter
DEFAULT_VALUE_KIND	NUMBER(8)	Integer indicating kind of value (Literal, Expression, Evaluated Expression, Not Set)
DEFAULT_VALUE	VARCHAR2(4000)	Default value for parameter

1.3 Execution Auditing Views

Table 1–15 ALL_RT_AUDIT_EXECUTIONS

Column Name	Data Type	Description
EXECUTION_AUDIT_ID	NUMBER (22)	Internal primary key to audit_execution
PARENT_EXECUTION_AUDIT_ID	NUMBER (22)	Internal key to parent audit_execution
TOP_LEVEL_EXECUTION_AUDIT_ID	NUMBER (22)	Internal key to top-level audit_execution
EXECUTION_NAME	VARCHAR2 (64)	Name of the execution run
TASK_NAME	VARCHAR2 (64)	Name of the task executed
TASK_TYPE	VARCHAR2 (64)	Type of task executed (PL/SQL, ProcessFlow).
TASK_INPUT	CLOB	Input stream for the task
EXEC_LOCATION_UOID	VARCHAR2 (32)	UOID of the location where execution is performed
EXEC_LOCATION_NAME	VARCHAR2 (64)	Name of the location where execution is performed
EXEC_LOCATION_TYPE	VARCHAR2 (64)	Type of the location where execution is performed. (Runtime Platform, OEM)
EXEC_LOCATION_TYPE_VERSION	VARCHAR2 (64)	Version of the location where execution is performed
OBJECT_UOID	VARCHAR2 (32)	Client UOID of mapping executed

Table 1–15 (Cont.) ALL_RT_AUDIT_EXECUTIONS

Column Name	Data Type	Description
OBJECT_NAME	VARCHAR2 (64)	Name of mapping executed
OBJECT_TYPE	VARCHAR2 (64)	Type of mapping executed
OBJECT_LOCATION_UOID	VARCHAR2 (32)	Location UOID where mapping deployed
OBJECT_LOCATION_NAME	VARCHAR2 (64)	Location name where mapping deployed
OBJECT_LOCATION_TYPE	VARCHAR2 (64)	Location type where mapping deployed
OBJECT_LOCATION_TYPE_VERSION	VARCHAR2 (64)	Location version where mapping deployed
RETURN_RESULT	VARCHAR2 (64)	FAILURE, OK, OK_WITH_WARNINGS, or OK_ WITH_ERRORS
RETURN_CODE	NUMBER (10)	<0: Failure >= 0: Success
EXECUTION_AUDIT_STATUS	VARCHAR2	INACTIVE, BUSY, READY, or COMPLETE
ELAPSE_TIME	NUMBER (10)	Number of seconds elapsed
NUMBER_TASK_ERRORS	NUMBER (10)	Number of errors detected
NUMBER_TASK_WARNINGS	NUMBER (10)	Number of warnings detected
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username
UPDATED_ON	DATE	The time the audit data was updated
UPDATED_BY	VARCHAR2 (30)	Database username

Table 1–16 ALL_RT_AUDIT_EXECUTION_PARAMS

Column Name	Data Type	Description
PARAMETER_AUDIT_ID	NUMBER (22)	Internal primary key to audit_execution_param
EXECUTION_AUDIT_ID	NUMBER (22)	Internal key to audit_execution
CUSTOM_PARAMETER_UOID	VARCHAR2 (32)	UOID of custom parameter
PARAMETER_NAME	VARCHAR2 (64)	Name of parameter
PARAMETER_TYPE	VARCHAR2(4000)	BOOLEAN, CHAR, DATE, FLOAT, NUMBER, VARCHAR, VARCHAR2, OPERATING_MODE, or AUDIT_LEVEL
PARAMETER_KIND	VARCHAR2(4000)	SYSTEM or CUSTOM
PARAMETER_MODE	VARCHAR2(4000)	IN, OUT, or INOUT
VALUE_KIND	VARCHAR2 (12)	INPUT VALUE or OUTPUT VALUE
VALUE	VARCHAR2 (4000)	Character representation of parameter value

Table 1–17 ALL_RT_AUDIT_EXEC_MESSAGES

Column Name	Data Type	Description
MESSAGE_AUDIT_ID	NUMBER (22)	Internal key to audit_exec_message. Primary key when used with message_line_number
EXECUTION_AUDIT_ID	NUMBER (22)	Internal key to audit_execution
MESSAGE_SEVERITY	VARCHAR2	INFORMATIONAL, WARNING, ERROR, or RECOVERY

Table 1–17 (Cont.) ALL_RT_AUDIT_EXEC_MESSAGES

Column Name	Data Type	Description
MESSAGE_LINE_NUMBER	NUMBER (10)	1 for single line messages >0 for multiple line messages (Forms primary key when used with message_audit_id))
MESSAGE_TEXT	VARCHAR2 (4000)	Plain_text or nls_key
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username

Table 1–18 ALL_RT_AUDIT_EXEC_FILES

Column Name	Data Type	Description
FILE_AUDIT_ID	NUMBER (22)	Internal primary key to audit_exec_file
EXECUTION_AUDIT_ID	NUMBER (22)	Internal key to audit_execution
FILE_TYPE	VARCHAR2 (64)	Type of the file
FILE_TEXT	CLOB	Content of the file
FORMAT	VARCHAR2	TEXT or HTML
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username

Table 1–19 ALL_RT_AUDIT_MAP_RUNS

Column Name	Data Type	Description
MAP_RUN_ID	NUMBER (22)	Internal primary key to audit_map_run
EXECUTION_AUDIT_ID	NUMBER (22)	Internal key to audit_execution
MAP_UOID	VARCHAR2 (255)	UOID of the mapping
MAP_NAME	VARCHAR2 (80)	Name of the mapping
MAP_TYPE	VARCHAR2 (30)	PLSQLMap or SQLLoaderControlFile
START_TIME	DATE	The time the mapping started
END_TIME	DATE	The time the mapping ended
ELAPSE_TIME	NUMBER (10)	Number of seconds elapsed
RUN_STATUS	VARCHAR2 (8)	RUNNING, FAILURE or COMPLETE FAILURE: This status means that the mapping run was completed with errors. COMPLETE: This status means that the mapping run has been completed successfully. However, when you specify a limit on the maximum number of errors that can be generated during the mapping run, if this limit is exceeded, the mapping run is stopped with the status COMPLETE.
PHYSICAL_NAME	VARCHAR2 (80)	Full hierarchic name of .dat file for a SQL*Loader run
LOAD_DATE	VARCHAR2 (30)	Load date for a SQL*Loader run
LOAD_TIME	VARCHAR2 (30)	Load time for a SQL*Loader run

Table 1–19 (Cont.) ALL_RT_AUDIT_MAP_RUNS

Column Name	Data Type	Description
NUMBER_ERRORS	NUMBER (10)	Number of errors detected
NUMBER_RECORDS_SELECTED	NUMBER (10)	Number of records selected from source tables
NUMBER_RECORDS_INSERTED	NUMBER (10)	Number of records inserted into target tables
NUMBER_RECORDS_UPDATED	NUMBER (10)	Number of records updated in target tables
NUMBER_RECORDS_DELETED	NUMBER (10)	Number of records deleted in target tables
NUMBER_RECORDS_DISCARDED	NUMBER (10)	Number of records discarded in SQL*Loader run
NUMBER_RECORDS_MERGED	NUMBER (10)	Number of records merged in target tables
NUMBER_RECORDS_CORRECTED	NUMBER (10)	Number of records corrected in target tables
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username
UPDATED_ON	DATE	The time the audit data was updated
UPDATED_BY	VARCHAR2 (30)	Database username

Table 1–20 ALL_RT_AUDIT_MAP_RUN_SOURCES

Column Name	Data Type	Description
MAP_RUN_ID	NUMBER (22)	Internal key to audit_map_run
SOURCE_NAME	VARCHAR2 (2000)	Name of mapping operator representing source table
SOURCE_DBLINK	VARCHAR2 (2000)	Name of database link for mapping operator representing source table

Table 1–21 ALL_RT_AUDIT_MAP_RUN_TARGETS

Column Name	Data Type	Description
MAP_RUN_ID	NUMBER (22)	Internal key to audit_map_run
TARGET_NAME	VARCHAR2 (2000)	Name of mapping operator representing target table

Table 1–22 ALL_RT_AUDIT_STEP_RUNS

Column Name	Data Type	Description
STEP_ID	NUMBER (22)	Internal primary key to audit_step_run
MAP_RUN_ID	NUMBER (22)	Internal key to audit_map_run
MAP_STEP	NUMBER (22)	Step number 0 or 1 For a PL/SQL mapping, this number is normally 0 for a set-based run, and 1 for a row-based, or row-based-target run
STEP_NAME	VARCHAR2 (80)	Name of the mapping for a set-based run, or the name of a mapping object for a set-based or set-based-target run
STEP_TYPE	VARCHAR2 (18)	Set-based, Row-based or Row-based target
START_TIME	DATE	The time the mapping step started
END_TIME	DATE	The time the mapping step ended

Table 1–22 (Cont.) ALL_RT_AUDIT_STEP_RUNS

Column Name	Data Type	Description
ELAPSE_TIME	NUMBER (10)	Number of seconds taken
RUN_STATUS	VARCHAR2 (8)	RUNNING or COMPLETE
NUMBER_ERRORS	NUMBER (10)	Number of errors detected
NUMBER_RECORDS_SELECTED	NUMBER (10)	Number of records selected from source tables
NUMBER_RECORDS_INSERTED	NUMBER (10)	Number of records inserted into target tables
NUMBER_RECORDS_UPDATED	NUMBER (10)	Number of records updated in target tables
NUMBER_RECORDS_DELETED	NUMBER (10)	Number of records deleted in target tables
NUMBER_RECORDS_DISCARDED	NUMBER (10)	Number of records discarded in a SQL*Loader run
NUMBER_RECORDS_MERGED	NUMBER (10)	Number of records merged in target tables
NUMBER_RECORDS_CORRECTED	NUMBER (10)	Number of records corrected in target tables
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username
UPDATED_ON	DATE	The time the audit data was updated
UPDATED_BY	VARCHAR2 (30)	Database username

Table 1–23 ALL_RT_AUDIT_STEP_RUN_SOURCES

Column Name	Data Type	Description
STEP_ID	NUMBER (22)	Internal key to audit_step_run
MAP_RUN_ID	NUMBER (22)	Internal key to audit_map_run
MAP_STEP	NUMBER (22)	Step number 0 or 1 For a PL/SQL mapping, this number is normally 0 for a set-based run, and 1 for a row-based, or row-based-target run
SOURCE_NAME	VARCHAR2 (2000)	Name of mapping operator representing source table
SOURCE_DBLINK	VARCHAR2 (2000)	Name of database link for mapping operator representing source table

Table 1–24 ALL_RT_AUDIT_STEP_RUN_TARGETS

Column Name	Data Type	Description
STEP_ID	NUMBER (22)	Internal key to audit_step_run
MAP_RUN_ID	NUMBER (22)	Internal key to audit_map_run
MAP_STEP	NUMBER (22)	Step number 0 or 1 For a PL/SQL mapping, this number is normally 0 for a set-based run, and 1 for a row-based, or row-based-target run
TARGET_NAME	VARCHAR2 (2000)	Name of mapping operator representing target table

Table 1–25 ALL_RT_AUDIT_MAP_RUN_ERRORS

Column Name	Data Type	Description
RUN_ERROR_ID	NUMBER (22)	Internal primary key for map_run_error
STEP_ID	NUMBER (22)	Internal key to audit_step_run
MAP_RUN_ID	NUMBER (22)	Internal key to audit_map_run
MAP_STEP	NUMBER (22)	Step number 0 or 1 For a PL/SQL mapping, this number is normally 0 for a set-based run, and 1 for a row-based, or row-based-target run.
CURSOR_ROWKEY	NUMBER (22)	Value identifying row returned by cursor. This is 0 for errors in a set-based run.
RUN_ERROR_NUMBER	NUMBER (10)	Message number
RUN_ERROR_MESSAGE	VARCHAR2 (2000)	Message text
TARGET_NAME	VARCHAR2 (80)	Name of mapping operator representing target table
TARGET_COLUMN	VARCHAR2 (80)	Column name, or '*' if not known or not applicable
STATEMENT	VARCHAR2 (2000)	Value such as INSERT or BATCH INSERT, or a PL/SQL statement
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username
UPDATED_ON	DATE	The time the audit data was updated
UPDATED_BY	VARCHAR2 (30)	Database username

Table 1–26 ALL_RT_AUDIT_MAP_RUN_TRACE

Column Name	Data Type	Description
TRACE_ID	NUMBER (22)	Internal primary key for map_run_trace
MAP_RUN_ID	NUMBER (22)	Internal key to audit_map_run
MAP_STEP	NUMBER (22)	Step number 0 or 1 For a PL/SQL mapping, this number is normally 0 for a set-based run, and 1 for a row-based, or row-based-target run
CURSOR_ROWKEY	NUMBER (22)	Value identifying error row returned by cursor. This is 0 for set-based run
TYPE	VARCHAR2 (30)	NEW for trace or ERROR for error
ROLE	VARCHAR2 (30)	S for source or T for target
ACTION	VARCHAR2 (30)	Value such as SELECT or a PL/SQL statement
TABLE_NAME	VARCHAR2 (80)	Name of mapping operator representing source/target table
CREATED_ON	DATE	The time the audit data was created
CREATED_BY	VARCHAR2 (30)	Database username
UPDATED_ON	DATE	The time the audit data was updated
UPDATED_BY	VARCHAR2 (30)	Database username

Table 1–27 ALL_RT_AUDIT_PROC_RUN_ERRORS

Column Name	Data Type	Description
RUN_ERROR_ID	NUMBER(22)	ID of the run error
MAP_RUN_ID	NUMBER(22)	ID of the map run
CURSOR_ROWKEY	NUMBER(22)	Rowkey of record returned by cursor when error reported
RUN_ERROR_NUMBER	NUMBER(10)	Error number
RUN_ERROR_MESSAGE	VARCHAR2(2000)	Error message
TARGET_NAME	VARCHAR2(2000)	Name of target
TARGET_COLUMN	VARCHAR2(80)	Name of target column
STATEMENT	VARCHAR2(2000)	Statement when error is reported
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 1–28 ALL_RT_AUDIT_STEP_RUN_STRUCTS

Column Name	Data Type	Description
STRUCT_ID	NUMBER(22)	ID of the structure
STEP_ID	NUMBER(22)	ID of the step in the map
MAP_RUN_ID	NUMBER(22)	ID of the run of the map
PARENT_OPERATOR_UOID	VARCHAR2(32)	UOID of the map-operator being audited
PARENT_OBJECT_UOID	VARCHAR2(32)	UOID of the parent object that the object is related to
PARENT_OBJECT_TYPE	VARCHAR2(30)	Type of the parent object
PARENT_OBJECT_LOCATION_UOID	VARCHAR2(32)	UOID of the location where the parent object has been deployed
PARENT_OBJECT_NAME	VARCHAR2(80)	Name of the parent object
OBJECT_UOID	VARCHAR2(32)	UOID of the object
OBJECT_TYPE	VARCHAR2(30)	Type of the object
OBJECT_LOCATION_UOID	VARCHAR2(32)	UOID of the location where the object has been deployed
OBJECT_NAME	VARCHAR2(80)	Name of the object
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2

Public Views for the Design Environment

The Warehouse Builder provides a set of pre-built views for both the design and runtime environments. These views are called the Warehouse Builder public views and are the API alternative to using the Repository Browser described in the *Oracle Warehouse Builder Data Modeling, ETL, and Data Quality Guide*.

Use these views to access to metadata and data stored in Warehouse Builder repositories.

You can access public views by logging in to SQL*Plus with default `workspace@set`. When you login to SQL*Plus, you can access public views (design-time public views or runtime public views) from your default workspace.

If you try to access public views from any workspace other than the default, then you must call: `WB_workspace_management.set_workspace`(<wksp_name>, <wksp_owner>). If you want to switch to a workspace other than the default one, then you can call the `WB_workspace_management.set_workspace` procedure. You must have the `ACCESS_PUBLICVIEW_BROWSER` system privilege to retrieve useful information from the public views. Otherwise, you will get "0 rows returned." You may need to ask the workspace owner or workspace admin to grant the system privilege `ACCESS_PUBLICVIEW_BROWSER`.

This chapter contains a catalog of the public views to the design environment.

2.1 Warehouse Builder Design Repository Public Views

The design repository contains all of the design metadata. Use these public views to access data about the design of your system. These views are used by Warehouse Builder Browser to provide metadata reporting.

General Model Views

Data Model Views

Business Intelligence Views

Note: In addition to the listed views, Warehouse Builder also contains the public view ALL_IV_TABLE_FUNC_PROPERTIES, which is now obsolete.

2.2 General Model Views

Table 2–1 ALL_IV_ALL_OBJECTS

Column Name	Data Type	Description
OBJECT_ID	NUMBER(9)	ID of the object
OBJECT_UOID	VARCHAR2(255)	UOID of the object
OBJECT_TYPE	VARCHAR2(4000)	Type of the object
OBJECT_NAME	VARCHAR2(4000)	Physical name of the object
BUSINESS_NAME	VARCHAR2(4000)	Business name of the object
CONTEXT_NAME	VARCHAR2(4000)	Name of the object, prefixed with its module's name, and project's name if it existed
DESCRIPTION	VARCHAR2(4000)	Description of the object
PARENT_OBJECT_ID	NUMBER(9)	Container object ID for the object. Container object could be a module, for example, for a dimension, or a table for a column
PARENT_OBJECT_TYPE	VARCHAR2(4000)	Type of the parent object

Table 2–1 (Cont.) ALL_IV_ALL_OBJECTS

Column Name	Data Type	Description
PARENT_OBJECT_NAME	VARCHAR2(4000)	Name of the parent object
IS_VALID	VARCHAR2(13)	Is the object valid? Applicable for only those objects that can be validated
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–2 ALL_IV_OBJECTS

Column Name	Data Type	Description
OBJECT_ID	NUMBER(9)	ID of the object (the difference between this view and ALL_IV_ALL_OBJECTS is that it includes all objects in this view, PLUS all archived snapshot objects (for MCM service))
OBJECT_TYPE	VARCHAR2(4000)	Type of the object
OBJECT_NAME	VARCHAR2(4000)	Physical name of the object
BUSINESS_NAME	VARCHAR2(4000)	Business name of the object
DESCRIPTION	VARCHAR2(4000)	Description of the object
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–3 ALL_IV_OBJECT_PROPERTIES

Column Name	Data Type	Description
OBJECT_ID	NUMBER(9)	ID of the object
OBJECT_TYPE	VARCHAR2(4000)	Type of the object
OBJECT_NAME	VARCHAR2(255)	Physical name of the object
PROPERTY_ID	NUMBER(9)	ID of the object's property
PROPERTY_NAME	VARCHAR2(255)	ID of the property name
PROPERTY_VALUE	VARCHAR2(4000)	Value of the property

Table 2–4 ALL_IV_MLS_OBJECTS

Column Name	Data Type	Description
OBJECT_ID	NUMBER(9)	ID of the object (covers the same set as ALL_IV_OBJECTS)
LANGUAGE_ID	VARCHAR2(255)	ID of the language (predefined internally by Warehouse Builder). To get language name, join this view with ALL_IV_SUPPORTED_LANGUAGES
BUSINESS_NAME	VARCHAR2(4000)	Business name of the object
DESCRIPTION	VARCHAR2(4000)	Description of the object

Table 2–5 ALL_IV_SUPPORTED_LANGUAGES

Column Name	Data Type	Description
LANGUAGE_ID	VARCHAR2(255)	ID of the language
LANGUAGE_NAME	VARCHAR2(64)	Name of the language
ISBASELANGUAGE	VARCHAR2(1)	Is it a base language (for example, EN or FR)

Table 2–6 ALL_IV_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project that this module belongs to
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the module
INFORMATION_SYSTEM_ NAME	VARCHAR2(255)	Physical name of the module
SCHEMA_ID	NUMBER(9)	ID of the module (repeated column, just to keep backward compatibility)
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
BUSINESS_NAME	VARCHAR2(4000)	Business name of the module
DESCRIPTION	VARCHAR2(4000)	Description of the module
STATUS	VARCHAR2(40)	Module status (dev, QA, prod)
IS_VALID	VARCHAR2(13)	Is this module valid
LOCATION_ID	NUMBER(9)	ID of the associated location for this module
LOCATION_NAME	VARCHAR2(255)	Name of associated location for this module
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–7 ALL_IV_PROJECTS

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
BUSINESS_NAME	VARCHAR2(4000)	Business name of the project
DESCRIPTION	VARCHAR2(4000)	Description of the project
VERSION_LABEL	VARCHAR2(255)	Version of the project
IS_VALID	VARCHAR2(13)	Is this project valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–8 ALL_IV_INFORMATION_SYSTEMS

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project that this module belongs to
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the module
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Physical name of the module
INFORMATION_SYSTEM_TYPE	VARCHAR2(4000)	Type of the module
BUSINESS_NAME	VARCHAR2(4000)	Business name of the module
DESCRIPTION	VARCHAR2(4000)	Description of the module
PRODUCT_TYPE	VARCHAR2(255)	Application type of the module (for example, Oracle apps, or File based apps)
SYSTEM_TYPE	VARCHAR2(255)	Type of system that holds this application (shown by PRODUCT_TYPE)
VERSION_LABEL	NUMBER(9)	Version of the module
VENDOR	VARCHAR2(40)	Vendor name
DATABASE_LINK	VARCHAR2(40)	Name of the database link that physical points to data storage of this module. It is meaningful only for database applications
INTEGRATOR_NAME	VARCHAR2(255)	The name of Warehouse Builder integrator component that is used to access external data source for the module
IS_VALID	VARCHAR2(13)	Is this module valid
LOCATION_ID	NUMBER(9)	ID of the associated location for this module
LOCATION_NAME	VARCHAR2(255)	Physical name of the associated location
STATUS	VARCHAR2(17)	Status (dev, QA, or prod)
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–9 ALL_IV_INSTALLATIONS

Column Name	Data Type	Description
INSTALLATION_ID	NUMBER(9)	ID of the Warehouse Builder repository
INSTALLATION_NAME	VARCHAR2(255)	Physical name of the Warehouse Builder repository
BUSINESS_NAME	VARCHAR2(4000)	Business name of the Warehouse Builder repository
DESCRIPTION	VARCHAR2(4000)	Description of the Warehouse Builder repository
INSTALLED_VERSION	VARCHAR2(40)	Version of the Warehouse Builder repository
RELEASE	VARCHAR2(40)	Version of the Warehouse Builder Client
REPOSITORY_MODEL_VERSION	NUMBER(9)	Version of the Warehouse Builder model
PUBLIC_VIEW_VERSION	CHAR(5)	Version of the Warehouse Builder public views

Table 2–9 (Cont.) ALL_IV_INSTALLATIONS

Column Name	Data Type	Description
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–10 ALL_IV_FILE_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project that this module belongs to
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of this file module
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Physical name of this file module
BUSINESS_NAME	VARCHAR2(4000)	Business name of this file module
DESCRIPTION	VARCHAR2(4000)	Description of this file module
PRODUCT_TYPE	VARCHAR2(255)	Application type of the module (for example, Oracle apps, or File based apps)
SYSTEM_TYPE	VARCHAR2(255)	Type of system that holds this application (represented by PRODUCT_TYPE)
VERSION_LABEL	NUMBER(9)	Version of the module
VENDOR	VARCHAR2(40)	Vendor name
DATABASE_LINK	VARCHAR2(40)	Name of the database link that physical points to data storage of this module. It is meaningful only for database applications
DIRECTORY	VARCHAR2(4000)	Name of the directory this file module connects to
INTEGRATOR_NAME	VARCHAR2(255)	The name of Warehouse Builder integrator component that is used to access external file system for the module
IS_VALID	VARCHAR2(13)	Is this module valid
LOCATION_ID	NUMBER(9)	ID of the associated location for this module
LOCATION_NAME	VARCHAR2(255)	Physical name of the associated location
STATUS	VARCHAR2(17)	Status (dev, QA, or prod)
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–11 ALL_IV_GATEWAY_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project that this module belongs to
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of this module
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Physical name of this module
BUSINESS_NAME	VARCHAR2(4000)	Business name of this module
DESCRIPTION	VARCHAR2(4000)	Description of this module
PRODUCT_TYPE	VARCHAR2(255)	Application type of the module (for example, Oracle apps or File based apps)
SYSTEM_TYPE	VARCHAR2(255)	Type of system that holds this application (represented by PRODUCT_TYPE)
VERSION_LABEL	NUMBER(9)	Version of the module
VENDOR	VARCHAR2(40)	Vendor name
DATABASE_LINK	VARCHAR2(40)	Name of the database link that physical points to data storage of this module. It is meaningful only for database applications
INTEGRATOR_NAME	VARCHAR2(255)	The name of Warehouse Builder integrator component that is used to access external data system for the module
STRONG_TYPE_NAME	VARCHAR2(255)	Used to differentiate which gateway component being employed, for example, Informix or Sybase
IS_VALID	VARCHAR2(13)	Is this module valid
LOCATION_ID	NUMBER(9)	ID of the associated location for this module
LOCATION_NAME	VARCHAR2(255)	Physical name of the associated location
STATUS	VARCHAR2(17)	Status (dev, QA, or prod)
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–12 ALL_IV_PACKAGED_APPS_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project that this module belongs to
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of this module (basically, the views wraps Oracle Applications, SAP)
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Physical name of this module
BUSINESS_NAME	VARCHAR2(4000)	Business name of this module
DESCRIPTION	VARCHAR2(4000)	Description of this module
PRODUCT_TYPE	VARCHAR2(255)	Application type of the module (for example, Oracle apps ,or File based apps)

Table 2–12 (Cont.) ALL_IV_PACKAGED_APPS_MODULES

Column Name	Data Type	Description
SYSTEM_TYPE	VARCHAR2(255)	Type of system that holds this application (represented by PRODUCT_TYPE)
VERSION_LABEL	NUMBER(9)	Version of the module
VENDOR	VARCHAR2(40)	Vendor name
DATABASE_LINK	VARCHAR2(40)	Name of the database link that physical points to data storage of this module. It is meaningful only for database applications.
INTEGRATOR_NAME	VARCHAR2(255)	The name of Warehouse Builder integrator component that is used to access external data system for the module
IS_VALID	VARCHAR2(13)	Is this module valid
LOCATION_ID	NUMBER(9)	ID of the associated location for this module
LOCATION_NAME	VARCHAR2(255)	Physical name of the associated location
STATUS	VARCHAR2(17)	Status (dev, QA, or prod)
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–13 ALL_IV_PREDEFINED_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project that this module belongs to
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of this module (basically, the views wraps Oracle Pre-defined Transformations and Public Transformations)
INFORMATION_SYSTEM_ NAME	VARCHAR2(255)	Physical name of this module
BUSINESS_NAME	VARCHAR2(4000)	Business name of this module
DESCRIPTION	VARCHAR2(4000)	Description of this module
PRODUCT_TYPE	VARCHAR2(255)	Application type of the module (for example, Oracle apps or File based apps)
SYSTEM_TYPE	VARCHAR2(255)	Type of system that holds this application (represented by PRODUCT_TYPE)
VERSION_LABEL	NUMBER(9)	Version of the module
VENDOR	VARCHAR2(40)	Vendor name
DATABASE_LINK	VARCHAR2(40)	Name of the database link that physical points to data storage of this module. It is meaningful only for database applications
INTEGRATOR_NAME	VARCHAR2(255)	The name of Warehouse Builder integrator component that is used to access external data system for the module
IS_VALID	VARCHAR2(13)	Is this module valid

Table 2–13 (Cont.) ALL_IV_PREDEFINED_MODULES

Column Name	Data Type	Description
STATUS	VARCHAR2(17)	Status (dev, QA, or prod)
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–14 ALL_IV_PROCESS_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project to which this module belongs
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of this module (basically, the views wraps Oracle Process Flow Module)
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Physical name of this module
BUSINESS_NAME	VARCHAR2(4000)	Business name of this module
DESCRIPTION	VARCHAR2(4000)	Description of this module
PRODUCT_TYPE	VARCHAR2(255)	Application type of the module (for example, Oracle apps or File based apps)
SYSTEM_TYPE	VARCHAR2(255)	Type of system that holds this application (represented by PRODUCT_TYPE)
VERSION_LABEL	NUMBER(9)	Version of the module
VENDOR	VARCHAR2(40)	Vendor name
DATABASE_LINK	VARCHAR2(40)	Name of the database link that physical points to data storage of this module. It is meaningful only for database applications.
INTEGRATOR_NAME	VARCHAR2(255)	The name of Warehouse Builder integrator component that is used to access external data system for the module
IS_VALID	VARCHAR2(13)	Is this module valid
LOCATION_ID	NUMBER(9)	ID of the associated location for this module
LOCATION_NAME	VARCHAR2(255)	Physical name of the associated location
STATUS	VARCHAR2(17)	Status (dev, QA, or prod)
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–15 ALL_IV_WAREHOUSE_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project that this module belongs to
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of this module (basically, the views wraps Oracle Warehouse Module)
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Physical name of this module
BUSINESS_NAME	VARCHAR2(4000)	Business name of this module
DESCRIPTION	VARCHAR2(4000)	Description of this module
PRODUCT_TYPE	VARCHAR2(255)	Application type of the module (for example, Oracle apps or File based apps)
SYSTEM_TYPE	VARCHAR2(255)	Type of system that holds this application (represented by PRODUCT_TYPE)
VERSION_LABEL	NUMBER(9)	Version of the module
VENDOR	VARCHAR2(40)	Vendor name
DATABASE_LINK	VARCHAR2(40)	Name of the database link that physical points to data storage of this module
INTEGRATOR_NAME	VARCHAR2(255)	The name of Warehouse Builder integrator component that is used to access external data system for the module
IS_VALID	VARCHAR2(13)	Is this module valid
LOCATION_ID	NUMBER(9)	ID of the associated location for this module
LOCATION_NAME	VARCHAR2(255)	Physical name of the associated location
STATUS	VARCHAR2(17)	Status (dev, QA, or prod)
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–16 ALL_IV_BUSINESS_DEF_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
BUSINESS_NAME	VARCHAR2(1000)	Business name of the business definition module
DESCRIPTION	VARCHAR2(4000)	Description of the business definition module
FOR DISCOVERER	NUMBER	A value of 1 indicates a business definition module for Oracle Discoverer.
FOR_OBI	NUMBER	A value of 1 indicates a business definition module for Oracle Business Intelligence.
DATABASE_LINK	VARCHAR2(40)	Not applicable.

Table 2–16 (Cont.) ALL_IV_BUSINESS_DEF_MODULES

Column Name	Data Type	Description
VALID	CHAR(1)	Validation result of the business definition module.
IS_VALID	VARCHAR2(13)	Validation status of the module.
STATUS	VARCHAR2(40)	The module status (Development, Quality Assurance, or Production)
LOCATION_ID	NUMBER(9)	ID of the Location associated with the business definition module
LOCATION_NAME	VARCHAR2(255)	Name of the Location associated with the business definition module
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–17 ALL_IV_BUSINESS_PRES_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
MODULE_ID	NUMBER(9)	ID of the business presentation module
MODULE_NAME	VARCHAR2(255)	Name of the business presentation module
BUSINESS_NAME	VARCHAR2(1000)	Business name of the business presentation module
DESCRIPTION	VARCHAR2(4000)	Description of the business presentation module
PRODUCT_TYPE	VARCHAR2(255)	Type of product (Oracle BI Beans Application)
SYSTEM_TYPE	VARCHAR2(255)	Internal
VERSION_LABEL	NUMBER(9)	Internal
VENDOR	VARCHAR2(40)	Name of the vendor (Currently the vendor name is Oracle)
DATABASE_LINK	VARCHAR2(40)	Not applicable
INTEGRATOR_NAME	VARCHAR2(255)	Internal
IS_VALID	VARCHAR2(13)	Validation status of the module
STATUS	VARCHAR2(40)	The module status (Development, Quality Assurance, or Production)
LOCATION_ID	NUMBER(9)	Id of the location with which the business presentation module is associated
LOCATION_NAME	VARCHAR2(255)	Name of the location with which the business presentation module is associated
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–18 ALL_IV_CALENDAR_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
CALENDAR_MODULE_ID	NUMBER(9)	ID of the calendar module
CALENDAR_MODULE_NAME	VARCHAR2(255)	Name of the calendar module
BUSINESS_NAME	VARCHAR2(1000)	Business name of the calendar module
DESCRIPTION	VARCHAR2(4000)	Description of the calendar module
IS_VALID	VARCHAR2(13)	Is the calendar module valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–19 ALL_IV_CMIV_DEFINITIONS

Column Name	Data Type	Description
CMIV_ID	NUMBER(9)	ID of the CMI definition
NAME	VARCHAR2(255)	Name of the CMI definition
BUSINESS_NAME	VARCHAR2(1000)	Business name
MIVMODE	VARCHAR2(40)	Mode of CMI (SQL or XML File)
TYPE	VARCHAR2(40)	Type
TABLE_FILTER_SUPPORTED	VARCHAR2(1)	Flag on supporting table filter
VIEW_FILTER_SUPPORTED	VARCHAR2(1)	Flag on supporting view filter
SEQUENCE_FILTER_ SUPPORTED	VARCHAR2(1)	Flag on supporting sequence filter
TABLE_FKLEVEL_SUPPORTED	VARCHAR2(1)	Flag on supporting table foreign key level dependency
MULTI_TREE_SUPPORTED	VARCHAR2(1)	Flag on supporting multiple tree in business component navigation
REIMPORT_SUPPORTED	VARCHAR2(1)	Flag on supporting reimport
TEST_DB_LINK	VARCHAR2(255)	DB link used for testing the CMI definition
TESTDIRECTORY	VARCHAR2(255)	File directory used for testing the CMI definition
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–20 ALL_IV_CMIV_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the information system
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Name of the information system
BUSINESS_NAME	VARCHAR2(1000)	Business name of the module
DESCRIPTION	VARCHAR2(4000)	Description of the module
PRODUCT_TYPE	VARCHAR2(255)	Product type
SYSTEM_TYPE	VARCHAR2(255)	System type
VERSION_LABEL	NUMBER(9)	Version label
VENDOR	VARCHAR2(40)	Vendor
DATABASE_LINK	VARCHAR2(40)	Database link
INTEGRATOR_NAME	VARCHAR2(255)	Name of the integrator
IS_VALID	VARCHAR2(13)	Flag on valid module
STATUS	VARCHAR2(40)	Status
LOCATION_ID	NUMBER(9)	ID of the location to access data
LOCATION_NAME	VARCHAR2(255)	Name of the data location
METADATA_LOCATION_ID	NUMBER(9)	ID of the location to access metadata
METADATA_LOCATION_NAME	VARCHAR2(255)	Name of the metadata location
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–21 ALL_IV_DATA_RULE_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project to which the data rule module belongs
PROJECT_NAME	VARCHAR2(255)	Name of the project to which the data rule module belongs
SCHEMA_ID	NUMBER(9)	ID of the data rule module
SCHEMA_NAME	VARCHAR2(255)	Name of the data rule module
BUSINESS_NAME	VARCHAR2(1000)	Business name of the data rule module
DESCRIPTION	VARCHAR2(4000)	Description of the data rule module
STATUS	VARCHAR2(40)	Not used
IS_VALID	VARCHAR2(13)	Not used
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–22 ALL_IV_EXPERT_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the information system
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Name of the information system
BUSINESS_NAME	VARCHAR2(1000)	The business name of the expert module
DESCRIPTION	VARCHAR2(4000)	Description of the module
VERSION_LABEL	NUMBER(9)	The version for this module
IS_VALID	VARCHAR2(13)	Is this module valid
STATUS	VARCHAR2(40)	The status for this module
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–23 ALL_IV_PF_CORRECTED_MODULES

Column Name	Data Type	Description
PROFILE_ID	NUMBER(9)	ID of the profile
PROFILE_NAME	VARCHAR2(255)	Name of the profile
CORRECTED_MODULE_ID	NUMBER(9)	ID of the corrected module
CORRECTED_MODULE_NAME	VARCHAR2(255)	Name of the corrected module
BUSINESS_NAME	VARCHAR2(1000)	Business name of the corrected module
DESCRIPTION	VARCHAR2(4000)	Description of the corrected module
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–24 ALL_IV_SAP_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the information system
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Name of the information system
BUSINESS_NAME	VARCHAR2(1000)	Business name of the SAP module
DESCRIPTION	VARCHAR2(4000)	Description of the SAP module
PRODUCT_TYPE	VARCHAR2(255)	Product type
SYSTEM_TYPE	VARCHAR2(255)	System type

Table 2–24 (Cont.) ALL_IV_SAP_MODULES

Column Name	Data Type	Description
VERSION_LABEL	NUMBER(9)	Version label
VENDOR	VARCHAR2(40)	Vendor
INTEGRATOR_NAME	VARCHAR2(255)	Integrator name
IS_VALID	VARCHAR2(13)	Flag if the module is valid
STATUS	VARCHAR2(40)	Status
LOCATION_ID	NUMBER(9)	Data location ID
LOCATION_NAME	VARCHAR2(255)	Data location name
METADATA_LOCATION_ID	NUMBER(9)	Metadata location ID
METADATA_LOCATION_NAME	VARCHAR2(255)	Metadata location name
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–25 ALL_IV_TM_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
TM_ID	NUMBER(9)	ID of the transportable module
TM_NAME	VARCHAR2(255)	Name of the transportable module
BUSINESS_NAME	VARCHAR2(1000)	Business name of the transportable module
DESCRIPTION	VARCHAR2(4000)	Description of the transportable module
IS_VALID	VARCHAR2(13)	Whether the transportable module is valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user
CHARSET	VARCHAR2(40)	Not used
ORACLEHOME	VARCHAR2(40)	Not used
HOST	VARCHAR2(40)	The source database host name
DEFAULTPORT	NUMBER(9)	The access port of source database
SERVICE	VARCHAR2(40)	The database service name of source database
SID	VARCHAR2(40)	Not used

Table 2–26 ALL_IV_UDO_MODULES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the information system
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Name of the information system
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
MODULE_TYPE	VARCHAR2(255)	Type of the module
BUSINESS_NAME	VARCHAR2(1000)	Business name of the module
DESCRIPTION	VARCHAR2(4000)	Description of the module
STATUS	VARCHAR2(40)	Status of the module
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–27 ALL_IV_CMIV_VIEWS

Column Name	Data Type	Description
MIV_DEFINITION_ID	NUMBER(9)	ID of the CMI definition
MIV_DEFINITION	VARCHAR2(255)	Name of the CMI definition
MIV_VIEW_ID	NUMBER(9)	ID of the CMI View
MIV_VIEW	VARCHAR2(255)	Name of the CMI View
BUSINESS_NAME	VARCHAR2(1000)	Business name of the CMI view
VIEW_TYPE	VARCHAR2(40)	Type of the CMI view
IS_DEFAULT	VARCHAR2(1)	Flag if the view definition is the default
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–28 ALL_IV_FIRSTCLASS_OBJECTS

Column Name	Data Type	Description
OBJECT_ID	NUMBER(9)	ID of the object
OBJECT_NAME	VARCHAR2(255)	Name of the object
BUSINESS_NAME	VARCHAR2(1000)	Business name of the object
DESCRIPTION	VARCHAR2(4000)	Description of the object
CLASSNAME	VARCHAR2(255)	Internal type of the object
OBJECT_TYPE	VARCHAR2(4000)	Type of object

Table 2–28 (Cont.) ALL_IV_FIRSTCLASS_OBJECTS

Column Name	Data Type	Description
SCRIPTING_TYPE	VARCHAR2(4000)	Type of object exposed in scripting
OBJECT_UOID	VARCHAR2(255)	UOID of the object
OWNINGFOLDER	NUMBER	Folder that owns the object
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–29 ALL_IV_DB_FUNCTIONS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
FUNCTION_LIBRARY_ID	NUMBER(9)	ID of the function library
FUNCTION_LIBRARY_NAME	VARCHAR2(255)	Name of the function library
FUNCTION_ID	NUMBER(9)	ID of the function
FUNCTION_NAME	VARCHAR2(255)	Name of the function
BUSINESS_NAME	VARCHAR2(1000)	Business name of the function
DESCRIPTION	VARCHAR2(4000)	Description of the function
SIGNATURE	VARCHAR2(4000)	Signature of the function
FUNCTION_TYPE	CHAR(8)	Function or procedure
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–30 ALL_IV_EXT_OBJECT_PROPERTIES

Column Name	Data Type	Description
OBJECT_ID	NUMBER(9)	ID of the object
OBJECT_TYPE	VARCHAR2(4000)	Type of the object
OBJECT_NAME	VARCHAR2(255)	Name of the object
CONFIGURATION_ID	NUMBER	ID of the configuration
CONFIGURATION_NAME	VARCHAR2(255)	Name of the configuration
PROPERTY_GROUP_ID	NUMBER(9)	ID of the property group (UI folder)
PROPERTY_GROUP_NAME	VARCHAR2(255)	Name of the property group
PROPERTY_GROUP_NLS_KEY	VARCHAR2(255)	Key for NLS translation lookup of property group name

Table 2–30 (Cont.) ALL_IV_EXT_OBJECT_PROPERTIES

Column Name	Data Type	Description
PROPERTY_SET_ID	NUMBER(9)	ID for property set
PROPERTY_SET_CLASS_NAME	VARCHAR2(255)	Name of defining class (may be superclass of OBJECT_TYPE)
PROPERTY_SET_NAME	VARCHAR2(255)	Name of property set (Typically 'Default' unless property set visibility is dynamically controlled in the clients)
PROPERTY_USAGE_ID	NUMBER(9)	ID of property usage (join between property set and property)
PROPERTY_STORAGE_TYPE	VARCHAR2(7)	Storage type for property. One of CORE, LOGICAL, CONFIG, USERDEF. (Describes how property was initially defined)
PROPERTY_USAGE_TYPE	VARCHAR2(7)	Usage type for property. One of CORE, LOGICAL, CONFIG, USERDEF. (Describes how property is used in this context)
PROPERTY_NAME	VARCHAR2(255)	Name of property
PROPERTY_NLS_KEY	VARCHAR2(255)	Key for NLS translation lookup of property name
PROPERTY_DATATYPE_ID	NUMBER	ID of property data type
PROPERTY_DATATYPE_NAME	VARCHAR2(255)	Name of property data type
PROPERTY_AFFECTS_SIGNATURE	VARCHAR2(4000)	Do changes to the value of this property affect the semantic signature of the owning object (true, false)
PROPERTY_DYNAMIC_DEFAULT	VARCHAR2(4000)	Fully qualified name of java class providing the default value for this property
PROPERTY_IS_HIDDEN	VARCHAR2(4000)	Is this property hidden from the GUI (true, false)
PROPERTY_PUBLISH_TO_PUBLIC_API	VARCHAR2(4000)	Is information on this property published in OMB*Plus help/documentation (true, false)
PROPERTY_DEFAULT_VALUE	VARCHAR2(4000)	Default value for property (may not be correct if dynamic default is defined)
PROPERTY_VALUE_OWNER_ID	NUMBER(9)	ID of the direct property owner
PROPERTY_VALUE_ID	NUMBER(9)	ID of the property value
PROPERTY_VALUE_INDEX	NUMBER(9)	Ordinal position of the property (If property type is an array)
PROPERTY_VALUE_NAME	VARCHAR2(255)	Name of property value (For properties that are objects, such as SQL loader data files)
PROPERTY_VALUE	VARCHAR2(4000)	Current value of property
DESCRIPTION	VARCHAR2(4000)	Developer description of property (In English)
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.3 Data Model Views

Table 2–31 ALL_IV_ADVANCED_QUEUES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this queue belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
QUEUE_ID	NUMBER(9)	ID of this queue
QUEUE_NAME	VARCHAR2(255)	Physical name of this queue
BUSINESS_NAME	VARCHAR2(4000)	Business name of this queue
DESCRIPTION	VARCHAR2(4000)	Description of this queue
LOAD_TYPE_ID	NUMBER(9)	ID of the load type
LOAD_TYPE_NAME	VARCHAR2(255)	Name of the load type
QUEUE_TABLE_NAME	VARCHAR2(40)	Name of the queue table
IS_VALID	VARCHAR2(13)	Is this queue valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–32 ALL_IV_ATTR_GROUPS

Column Name	Data Type	Description
DATA_ENTITY_ID	NUMBER(9)	ID of the data entity this attribute group belongs to
DATA_ENTITY_TYPE	VARCHAR2(4000)	Type of the data entity
DATA_ENTITY_NAME	VARCHAR2(255)	Physical name of the data entity
ATTRIBUTE_GROUP_NAME	VARCHAR2(255)	Physical name of this attribute group
ATTRIBUTE_GROUP_ID	NUMBER(9)	ID of this attribute group
BUSINESS_NAME	VARCHAR2(4000)	Business name of this attribute group
DESCRIPTION	VARCHAR2(4000)	Description of this attribute group
ATTRIBUTE_GROUP_TYPE	VARCHAR2(40)	Type of attribute group
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–33 ALL_IV_ATTR_GROUP_ITEM_USES

Column Name	Data Type	Description
ATTRIBUTE_GROUP_ID	NUMBER(9)	ID of the attribute group that this data item belongs to
ATTRIBUTE_GROUP_NAME	VARCHAR2(255)	Name of the attribute group
DATA_ITEM_ID	NUMBER(9)	ID of this data item

Table 2–33 (Cont.) ALL_IV_ATTR_GROUP_ITEM_USES

Column Name	Data Type	Description
DATA_ITEM_TYPE	VARCHAR2(4000)	Type of this data item
DATA_ITEM_NAME	VARCHAR2(255)	Physical name of this data item
POSITION	NUMBER(9)	Position of this data item in the attribute group

Table 2–34 ALL_IV_CHECK_CONSTRAINTS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this check constraint belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
RELATION_ID	NUMBER(9)	ID of the relation entity this check constraint belongs to
RELATION_NAME	VARCHAR2(255)	Physical name of the relation entity
CONSTRAINT_ID	NUMBER(9)	ID of this check constraint
CONSTRAINT_NAME	VARCHAR2(255)	Physical name of this check constraint
BUSINESS_NAME	VARCHAR2(4000)	Business name of this check constraint
DESCRIPTION	VARCHAR2(4000)	Description of this check constraint
CONSTRAINT_TEXT	VARCHAR2(255)	Textual expression of this check constraint
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–35 ALL_IV_COLUMNS

Column Name	Data Type	Description
ENTITY_ID	NUMBER(9)	ID of the data entity this column belongs to
ENTITY_TYPE	VARCHAR2(4000)	Type of the data entity
ENTITY_NAME	VARCHAR2(255)	Physical name of the data entity
COLUMN_ID	NUMBER(9)	ID of this column
COLUMN_NAME	VARCHAR2(255)	Physical name of this column
BUSINESS_NAME	VARCHAR2(4000)	Business name of this column
DESCRIPTION	VARCHAR2(4000)	Description of this column
POSITION	NUMBER(9)	Position of this column in the data entity
DATA_TYPE	VARCHAR2(255)	Data type of this column
LENGTH	NUMBER(9)	Data length of this column
PRECISION	NUMBER(9)	Data precision of this column
SCALE	NUMBER(9)	Data scale of this column
UPDATED_ON	DATE	Update timestamp

Table 2–35 (Cont.) ALL_IV_COLUMNS

Column Name	Data Type	Description
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–36 ALL_IV_CONSTRAINTS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this constraint belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
RELATION_ID	NUMBER(9)	ID of the relational entity this constraint belongs to
RELATION_NAME	VARCHAR2(255)	Physical name of the relational entity
CONSTRAINT_ID	NUMBER(9)	ID of this constraint
CONSTRAINT_NAME	VARCHAR2(255)	Physical name of this constraint
CONSTRAINT_TYPE	VARCHAR2(21)	Type of this constraint (check, primary, foreign key)
BUSINESS_NAME	VARCHAR2(4000)	Business name of this constraint
DESCRIPTION	VARCHAR2(4000)	Description of this constraint
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–37 ALL_IV_CUBES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this cube belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
CUBE_ID	NUMBER(9)	ID of this cube
CUBE_NAME	VARCHAR2(255)	Physical name of this cube
BUSINESS_NAME	VARCHAR2(4000)	Business name of this cube
DESCRIPTION	VARCHAR2(4000)	Description of this cube
IS_VALID	VARCHAR2(13)	Is this cube valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user
AUTOSOLVE	CHAR(1)	Sets the flag to say whether to measure is Auto Solve
AWNAME	VARCHAR2(255)	The analytical workspace name where the cube is implemented
AWOBJECTNAME	VARCHAR2(4000)	The analytical workspace object name

Table 2–37 (Cont.) ALL_IV_CUBES

Column Name	Data Type	Description
AWTABLESPACENAME	VARCHAR2(255)	The analytical workspace tablespace name
COMPRESSED	CHAR(1)	Flag to check whether the cube is compressed
CREATEBITMAPS	CHAR(1)	Flag to check whether to create a bitmap for the cube
CREATECONSTRAINTS	CHAR(1)	Flag to check whether to create a constraint for the cube
IMPLEMENTATION	VARCHAR2(255)	Whether the storage of a cube is AW or Relational
LOADPOLICY	VARCHAR2(255)	Not applicable for the current release
OLAPUSERVISIBLE	CHAR(1)	Flag to check whether the Cube is visible to OLAP end user
STORAGEPROPERTYTYPE	VARCHAR2(255)	The storage of a cube can be AW or Relational
USEGLOBALINDEX	CHAR(1)	Whether to generate a composite for measure partition combination
PARTITIONHIERARCHY	NUMBER(9)	The hierarchy by which one should partition the cube
INSTALLEDMODULE	NUMBER(9)	The Oracle module to which the cube belongs
PARTITIONLEVEL	NUMBER(9)	The level by which one should partition the cube
BINDINGFACT	NUMBER(9)	The element ID for binding fact table to the cube

Table 2–38 ALL_IV_CUBE_DIMENSIONS

Column Name	Data Type	Description
CUBE_ID	NUMBER(9)	ID of the cube this dimension has associated with
CUBE_NAME	VARCHAR2(255)	Physical name of the cube
DIMENSION_ID	NUMBER(9)	ID of this dimension
DIMENSION_NAME	VARCHAR2(255)	Physical name of this dimension
DIMENSION_ALIAS	VARCHAR2(255)	Alias of this dimension

Table 2–39 ALL_IV_CUBE_MEASURES

Column Name	Data Type	Description
CUBE_ID	NUMBER(9)	ID of the cube this measure belongs to
CUBE_NAME	VARCHAR2(255)	Physical name of the cube
MEASURE_ID	NUMBER(9)	ID of this measure
MEASURE_NAME	VARCHAR2(255)	Physical name of this measure
BUSINESS_NAME	VARCHAR2(4000)	Business name of this measure
DESCRIPTION	VARCHAR2(4000)	Description of this measure
POSITION	NUMBER(9)	Position of this measure within the cube
DATA_TYPE	VARCHAR2(255)	Data type of this measure
LENGTH	NUMBER(9)	Data length of this measure
PRECISION	NUMBER(9)	Data precision of this measure
SCALE	NUMBER(9)	Data scale of this measure

Table 2–39 (Cont.) ALL_IV_CUBE_MEASURES

Column Name	Data Type	Description
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user
TABLE_ID	NUMBER(9)	ID of the table
TABLE_NAME	VARCHAR2(255)	Name of the table

Note: The view ALL_IV_CUBE_MEASURE_DIM_USES is redundant. It can be created by joining ALL_IV_PF_CORRECTED_MODULES and ALL_IV_SAP_MODULES. It is scheduled for obsolescence.

Table 2–40 ALL_IV_CUBE_MEASURE_DIM_USES

Column Name	Data Type	Description
CUBE_ID	NUMBER(9)	ID of the cube
CUBE_NAME	VARCHAR2(255)	Physical name of the cube
MEASURE_ID	NUMBER(9)	ID of the measure belonging to this cube
MEASURE_NAME	VARCHAR2(255)	Physical name of the measure
DIMENSION_ID	NUMBER	ID of the dimension associated with this cube
DIMENSION_NAME	VARCHAR2(255)	Physical name of the dimension
DIMENSION_ALIAS	VARCHAR2(255)	Alias of the dimension

Table 2–41 ALL_IV_DIMENSIONS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this dimension belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
DIMENSION_ID	NUMBER(9)	ID of this dimension
DIMENSION_NAME	VARCHAR2(255)	Physical name of this dimension
BUSINESS_NAME	VARCHAR2(4000)	Business name of this dimension
DESCRIPTION	VARCHAR2(4000)	Description of this dimension
IS_VALID	VARCHAR2(13)	Is this dimension valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user
AWNAME	VARCHAR2(255)	The analytical workspace name where the dimension is implemented
AWOBJECTNAME	VARCHAR2(4000)	The analytical workspace object name

Table 2–41 (Cont.) ALL_IV_DIMENSIONS

Column Name	Data Type	Description
AWTABLESPACENAME	VARCHAR2(255)	The analytical workspace tablespace name
CREATECONSTRAINTS	CHAR(1)	Flag to check whether to create a constraint for the dimension
IMPLEMENTATION	VARCHAR2(255)	Type of implementation of the dimension (The storage of a dimension can be AW or Relational)
LOADPOLICY	VARCHAR2(255)	Data policy for loading dimension where Warehouse Builder mapping code relies on database constraints to detect the orphans (level records without parent)
OLAPPRIMARYSORTORDER	VARCHAR2(255)	The primary sorting order for dimension data in OLAP service
OLAPSECONDARYSORTORDER	VARCHAR2(255)	The secondary sorting order for dimension data in OLAP service
OLAPTYPE	VARCHAR2(255)	Type of OLAP Dimension Normal or Time
OLAPUSERVISIBLE	CHAR(1)	Whether the dimension is visible to OLAP end user
OWBTYPE	VARCHAR2(255)	Type of Dimension Normal or Time
REMOVEPOLICY	VARCHAR2(255)	Orphan Management (not applicable for the current release)
SLOWLYCHANGINGTYPE	NUMBER(9)	Slowly changing policy to be applied on the dimension
STORAGEPROPERTYTYPE	VARCHAR2(255)	The storage of a cube can be AW or Relational
VALUEBASED	CHAR(1)	The flag to define a value based hierarchy, applicable for AW only
DEFAULTDISPLAYHIERARCHY	NUMBER(9)	The hierarchy is set as default display hierarchy
INSTALLEDMODULE	NUMBER(9)	The Oracle module to which the cube belongs
TIMEDIMPOPULATIONMAP	NUMBER(9)	The element ID of map for a time dimension
DIMENSIONKEYSEQUENCE	NUMBER(9)	Element ID for sequence to generate the dimension key

Table 2–42 ALL_IV_DIM_HIERARCHIES

Column Name	Data Type	Description
DIMENSION_ID	NUMBER(9)	ID of the dimension this hierarchy belongs to
DIMENSION_NAME	VARCHAR2(255)	Physical name of this dimension
HIERARCHY_ID	NUMBER(9)	ID of this hierarchy
HIERARCHY_NAME	VARCHAR2(255)	Physical name of this hierarchy
BUSINESS_NAME	VARCHAR2(1000)	Business name of this hierarchy
DESCRIPTION	VARCHAR2(4000)	Description of this hierarchy
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–43 ALL_IV_DIM_HIERARCHY_LEVELS

Column Name	Data Type	Description
LEVEL_USE_ID	NUMBER(9)	ID of the level relationship that this level and this parent level participates in
LEVEL_USE_NAME	VARCHAR2(255)	Level name used on Hierarchy
HIERARCHY_ID	NUMBER(9)	ID of the hierarchy that this level and this parent level belongs to
HIERARCHY_NAME	VARCHAR2(255)	Physical name of the hierarchy
LEVEL_ID	NUMBER(9)	ID of this level
LEVEL_NAME	VARCHAR2(255)	Physical name of this level
LEVEL_DESCRIPTION	VARCHAR2(4000)	Description of this level
PARENT_LEVEL_ID	NUMBER(9)	ID of this parent level
PARENT_LEVEL_ NAME	VARCHAR2(255)	Physical name of this parent level
POSITION	NUMBER(9)	Position of this level

Table 2–44 ALL_IV_DIM_LEVELS

Column Name	Data Type	Description
DIMENSION_ID	NUMBER(9)	ID of the dimension this level belongs to
DIMENSION_NAME	VARCHAR2(255)	Physical name of the dimension
LEVEL_ID	NUMBER(9)	ID of this level
LEVEL_NAME	VARCHAR2(255)	Physical name of this level
BUSINESS_NAME	VARCHAR2(4000)	Business name of this level
DESCRIPTION	VARCHAR2(4000)	Description of this level
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–45 ALL_IV_DIM_LEVEL_ATTRIBUTES

Column Name	Data Type	Description
LEVEL_ID	NUMBER(9)	ID of the level this attribute belongs to
LEVEL_NAME	VARCHAR2(255)	Physical name of the level
ATTRIBUTE_ID	NUMBER(9)	ID of this attribute
ATTRIBUTE_NAME	VARCHAR2(255)	Physical name of this attribute
DIMENSION_ATTRIBUTE	VARCHAR2(255)	Name of the dimension attribute that the level attribute implements
BUSINESS_NAME	VARCHAR2(1000)	Business name of this attribute
DESCRIPTION	VARCHAR2(4000)	Description of this attribute
POSITION	NUMBER(9)	Position of this attribute within the level
DATA_TYPE	VARCHAR2(255)	Data type of this attribute

Table 2–45 (Cont.) ALL_IV_DIM_LEVEL_ATTRIBUTES

Column Name	Data Type	Description
LENGTH	NUMBER(9)	Data length of this attribute
PRECISION	NUMBER(9)	Data precision of this attribute
SCALE	NUMBER(9)	Data scale of this attribute
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–46 ALL_IV_EXTERNAL_COLUMNS

Column Name	Data Type	Description
ENTITY_ID	NUMBER(9)	ID of the external table that this column belongs to
ENTITY_NAME	VARCHAR2(255)	Name of the external table
COLUMN_ID	NUMBER(9)	ID of this column
COLUMN_NAME	VARCHAR2(255)	Physical name of this column
BUSINESS_NAME	VARCHAR2(4000)	Business name of this column
DESCRIPTION	VARCHAR2(4000)	Description of this column
POSITION	NUMBER(9)	Position of this column within the external table
DATA_TYPE	VARCHAR2(255)	Data type of this column
LENGTH	NUMBER(9)	Data length of this column
PRECISION	NUMBER(9)	Data precision of this column
SCALE	NUMBER(9)	Data scale of this column
SOURCE_FIELD_ID	NUMBER(9)	ID of the field that this column maps to
SOURCE_FIELD_NAME	VARCHAR2(255)	Physical name of the source field
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–47 ALL_IV_EXTERNAL_TABLES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this external table belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
LOCATION_ID	NUMBER(9)	ID of the location where the module is deployed to
LOCATION_NAME	VARCHAR2(255)	Physical name of the location
TABLE_ID	NUMBER(9)	ID of the external table
TABLE_NAME	VARCHAR2(255)	Physical name of the external table
BUSINESS_NAME	VARCHAR2(4000)	Business name of the external table

Table 2–47 (Cont.) ALL_IV_EXTERNAL_TABLES

Column Name	Data Type	Description
DESCRIPTION	VARCHAR2(4000)	Description of the external table
SOURCE_RECORD_ID	NUMBER(9)	ID of the record that this external table maps to
SOURCE_RECORD_NAME	VARCHAR2(255)	Physical name of the source record
SOURCE_FILE_NAME	VARCHAR2(255)	Physical name of the file that this source record belongs to
ACCESS_PARAMETERS	VARCHAR2(4000)	Expression for parameters that are used to access the source record
IS_VALID	VARCHAR2(13)	Is this external table valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–48 ALL_IV_FOREIGN_KEYS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module that this foreign key belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
ENTITY_ID	NUMBER(9)	ID of the data entity this foreign key belongs to
ENTITY_NAME	VARCHAR2(255)	Physical name of the data entity
ENTITY_TYPE	VARCHAR2(4000)	Type of the data type (for example, table, view)
FOREIGN_KEY_ID	NUMBER(9)	ID of this foreign key
FOREIGN_KEY_NAME	VARCHAR2(255)	Physical name of this foreign key
BUSINESS_NAME	VARCHAR2(4000)	Business name of this foreign key
DESCRIPTION	VARCHAR2(4000)	Description of this foreign key
KEY_ID	NUMBER(9)	ID of the associated key for this foreign key
KEY_NAME	VARCHAR2(255)	Physical name of the key
IS_DISABLED	CHAR(1)	Is this foreign key disabled
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–49 ALL_IV_KEYS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this key belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of this module
ENTITY_ID	NUMBER(9)	ID of the data entity this key belongs to
ENTITY_NAME	VARCHAR2(255)	Physical name of the data entity

Table 2–49 (Cont.) ALL_IV_KEYS

Column Name	Data Type	Description
ENTITY_TYPE	VARCHAR2(4000)	Type of the data entity (for example, table, view)
KEY_ID	NUMBER(9)	ID of this key
KEY_NAME	VARCHAR2(255)	Physical name of this key
BUSINESS_NAME	VARCHAR2(4000)	Business of this key
DESCRIPTION	VARCHAR2(4000)	Description of this key
IS_PRIMARY	VARCHAR2(9)	Is this key primary key
IS_DISABLED	CHAR(1)	Is this key disabled
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–50 ALL_IV_KEY_COLUMN_USES

Column Name	Data Type	Description
KEY_ID	NUMBER(9)	ID of the key that this column is associated with
KEY_NAME	VARCHAR2(255)	Physical name of the key
KEY_TYPE	VARCHAR2(11)	Type of the key (primary, unique, foreign)
COLUMN_ID	NUMBER(9)	ID of this column
COLUMN_NAME	VARCHAR2(255)	Physical name of this column
POSITION	NUMBER(9)	Position of this column with the key

Table 2–51 ALL_IV_MATERIALIZED_VIEWS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this materialized view belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
VIEW_ID	NUMBER(9)	ID of this materialized view
VIEW_NAME	VARCHAR2(255)	Physical name of this materialized view
BUSINESS_NAME	VARCHAR2(4000)	Business name of this materialized view
DESCRIPTION	VARCHAR2(4000)	Description of this materialized view
QUERY_TEXT	VARCHAR2(4000)	Textual expression of query statement for this materialized view
IS_VALID	VARCHAR2(13)	Is this materialized view valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–52 ALL_IV_OBJECT_TYPES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this object type belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
FOLDER_ID	NUMBER(9)	ID of the folder this object type belongs to
FOLDER_NAME	VARCHAR2(255)	Physical name of the folder
OBJECT_TYPE_ID	NUMBER(9)	ID of this object type
OBJECT_TYPE_NAME	VARCHAR2(255)	Physical name of this object type
BUSINESS_NAME	VARCHAR2(4000)	Business name of this object
DESCRIPTION	VARCHAR2(4000)	Description of this object type
TYPE	VARCHAR2(40)	Type of this object type
IS_VALID	VARCHAR2(13)	Is this object type valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–53 ALL_IV_RECORD_FIELDS

Column Name	Data Type	Description
FIRSTCLASS_OBJECT_ID	NUMBER(9)	ID of the first class object that this record field belongs to (normally, this ID will be the same as the following relational ID)
FRISTCLASS_OBJECT_NAME	VARCHAR2(255)	Physical name of the first class object
RELATION_ID	NUMBER(9)	ID of the relational entity this record field belongs to
RELATION_NAME	VARCHAR2(255)	Physical name of the relational entity
RECORDFIELD_ID	NUMBER(9)	ID of this record field
RECORDFIELD_NAME	VARCHAR2(255)	Physical name of this record field
BUSINESS_NAME	VARCHAR2(4000)	Business name of this record field
DESCRIPTION	VARCHAR2(4000)	Description of this record field
POSITION	NUMBER(9)	Position of this record field
DATA_TYPE	VARCHAR2(255)	Data type of this record field
LENGTH	NUMBER(9)	Data length of this record field
PRECISION	NUMBER(9)	Data precision of this record field
SCALE	NUMBER(9)	Data scale of this record field
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–54 ALL_IV_RELATIONS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this relational entity belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
RELATION_ID	NUMBER(9)	ID of this relational entity
RELATION_NAME	VARCHAR2(255)	Physical name of this relational entity
RELATION_TYPE	VARCHAR2(16)	Type of this relational entity (such as table, view, sequence and materialized view)
BUSINESS_NAME	VARCHAR2(4000)	Business name of this relational entity
DESCRIPTION	VARCHAR2(4000)	Description of this relational entity
IS_VALID	VARCHAR2(13)	Is this relational entity valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–55 ALL_IV_SEQUENCES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this sequence belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module the sequence belongs to
SEQUENCE_ID	NUMBER(9)	ID of the sequence
SEQUENCE_NAME	VARCHAR2(255)	Physical name of the sequence
BUSINESS_NAME	VARCHAR2(4000)	Business name of the sequence
DESCRIPTION	VARCHAR2(4000)	Description of the sequence
IS_VALID	VARCHAR2(13)	Is this sequence valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–56 ALL_IV_VIEWS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this view belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
VIEW_ID	NUMBER(9)	ID of this view
VIEW_NAME	VARCHAR2(255)	Physical name of this view
QUERY_TEXT	VARCHAR2(4000)	Textual expression of the query for this view
BUSINESS_NAME	VARCHAR2(4000)	Business name of this view
DESCRIPTION	VARCHAR2(4000)	Description of this view

Table 2–56 (Cont.) ALL_IV_VIEWS

Column Name	Data Type	Description
IS_VALID	VARCHAR2(13)	Is this view valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–57 ALL_IV_TABLES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this table belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
TABLE_ID	NUMBER(9)	ID of this table
TABLE_NAME	VARCHAR2(255)	Physical name of this table
BUSINESS_NAME	VARCHAR2(4000)	Business name of this table
DESCRIPTION	VARCHAR2(4000)	Description of this table
IS_VALID	VARCHAR2(13)	Is this table valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–58 ALL_IV_CALENDARS

Column Name	Data Type	Description
CALENDAR_MODULE_ID	NUMBER(9)	ID of the owning calendar module
CALENDAR_MODULE_NAME	VARCHAR2(255)	Name of the owning calendar module
CALENDAR_ID	NUMBER(9)	ID of the calendar
CALENDAR_NAME	VARCHAR2(255)	Name of the calendar
BUSINESS_NAME	VARCHAR2(1000)	Business name of the calendar
DESCRIPTION	VARCHAR2(4000)	Description of the calendar
IS_VALID	VARCHAR2(13)	Is the calendar valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–59 ALL_IV_VARRAYS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
VARRAY_ID	NUMBER(9)	ID of the varray
VARRAY_NAME	VARCHAR2(255)	Name of the varray
BUSINESS_NAME	VARCHAR2(1000)	Business name of the varray
DESCRIPTION	VARCHAR2(4000)	Description of the varray
IS_VALID	VARCHAR2(13)	Is the varray valid
BASE_ELEMENT_NAME	VARCHAR2(767)	Name of the base element
BASE_ELEMENT_PRECISION	NUMBER	Precision of the base element
BASE_ELEMENT_SCALE	NUMBER	Scale of the base element
BASE_ELEMENT_LENGTH	NUMBER	Length of the base element
ARRAY_LENGTH	NUMBER(9)	Number of elements in the varray
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–60 ALL_IV_SCHEMAS

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
BUSINESS_NAME	VARCHAR2(1000)	Business name of the schema
DESCRIPTION	VARCHAR2(4000)	Description of the schema
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the information system
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Name of the information system
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Create timestamp

Table 2–61 ALL_IV_PROCEDURES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
FUNCTION_LIBRARY_ID	NUMBER(9)	ID of the function library
FUNCTION_LIBRARY_NAME	VARCHAR2(255)	Name of the function library

Table 2–61 (Cont.) ALL_IV_PROCEDURES

Column Name	Data Type	Description
FUNCTION_ID	NUMBER(9)	ID of the procedure
FUNCTION_NAME	VARCHAR2(255)	Physical name of the procedure
BUSINESS_NAME	VARCHAR2(1000)	Business name of the procedure
DESCRIPTION	VARCHAR2(4000)	Description of the procedure
SIGNATURE	VARCHAR2(4000)	Signature of procedure
FUNCTION_TYPE	CHAR(9)	Type of the function
IS_VALID	VARCHAR2(13)	Is procedure valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–62 ALL_IV_REF_CURSORS

Column Name	Data Type	Description
LIBRARY_ID	NUMBER(9)	ID of the library
LIBRARY_NAME	VARCHAR2(255)	Name of the library
CURSOR_ID	NUMBER(9)	ID of the cursor
CURSOR_NAME	VARCHAR2(255)	Name of the cursor
BUSINESS_NAME	VARCHAR2(1000)	Business name of the cursor
DESCRIPTION	VARCHAR2(4000)	Description of the cursor
CURSOR_TYPE	VARCHAR2(40)	Type of the cursor
RETURN_RECORD_ID	NUMBER(9)	Ref Cursor Row Type Id
RETURN_RECORD_NAME	VARCHAR2(255)	Ref Cursor Row Type Name
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–63 ALL_IV_DIM_ATTRIBUTES

Column Name	Data Type	Description
DIMENSION_ID	NUMBER(9)	ID of the dimension
DIMENSION_NAME	VARCHAR2(255)	Name of the dimension
ATTRIBUTE_ID	NUMBER(9)	ID of the attribute
ATTRIBUTE_NAME	VARCHAR2(255)	Name of the attribute
BUSINESS_NAME	VARCHAR2(1000)	Business name of the attribute
DESCRIPTION	VARCHAR2(4000)	Description of the attribute
POSITION	NUMBER(9)	Position of the dimension attribute

Table 2–63 (Cont.) ALL_IV_DIM_ATTRIBUTES

Column Name	Data Type	Description
DATA_TYPE	VARCHAR2(255)	Data type of the dimension attribute
LENGTH	NUMBER(9)	Length for data types of the dimension attribute
PRECISION	NUMBER(9)	Precision for data types of the dimension attribute
SCALE	NUMBER(9)	Scale for data types of the dimension attribute
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–64 ALL_IV_DIM_ROLES

Column Name	Data Type	Description
DIMENSION_ID	NUMBER(9)	ID of the dimension
DIMENSION_NAME	VARCHAR2(255)	Name of the dimension
ROLE_ID	NUMBER(9)	ID of the role
ROLE_NAME	VARCHAR2(255)	Name of the role
BUSINESS_NAME	VARCHAR2(1000)	Business name of the role
DESCRIPTION	VARCHAR2(4000)	Description of the role
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–65 ALL_IV_TM_SCHEMAS

Column Name	Data Type	Description
TM_TABLESPACE_ID	NUMBER(9)	ID of the owning tablespace
TM_TABLESPACE_NAME	VARCHAR2(255)	Internal name of the owning tablespace
TM_SCHEMA_ID	NUMBER(9)	ID of the schema
TM_SCHEMA_NAME	VARCHAR2(255)	Internal unique name for the schema
TM_SCHEMA_UI_NAME	VARCHAR2(1002)	Schema name of the schema in source database. This may not be unique
BUSINESS_NAME	VARCHAR2(1000)	Business name of the schema
DESCRIPTION	VARCHAR2(4000)	Description of the schema
IS_VALID	VARCHAR2(13)	Not used
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–66 ALL_IV_TM_TABLESPACES

Column Name	Data Type	Description
TM_ID	NUMBER(9)	ID of the transportable module
TM_NAME	VARCHAR2(255)	Name of the transportable module
TM_TABLESPACE_ID	NUMBER(9)	Internal ID assigned to tablespace within the transportable module
TM_TABLESPACE_NAME	VARCHAR2(255)	Internal name assigned to tablespace within the transportable module. The internal tablespace name is unique
BUSINESS_NAME	VARCHAR2(1000)	The full specification of the tablespace. The format of this value is <Internal tablespace name>:<Source Host>:<Source Port>:<Source Service>:<Source Tablespace Name>
TM_TABLESPACE_UI_NAME	VARCHAR2(1000)	The source tablespace name that this tablespace represent. The tablespace UI name is not guaranteed to be unique
DESCRIPTION	VARCHAR2(4000)	Description of the tablespace
IS_VALID	VARCHAR2(13)	Not used
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user
EXPANDABLE	CHAR(1)	Not used
BLOCKSIZE	NUMBER(9)	Not used

Table 2–67 ALL_IV_CUBE_IMPLS

Column Name	Data Type	Description
IMPLEMENTATION_ID	NUMBER(9)	ID of this cube (this column will be updated in future)
ITEM_ID	NUMBER(9)	ID of the item belonging to this cube
ITEM_TYPE	VARCHAR2(18)	Type of item, either cube measures or cube dimension use (foreign keys pointing to dimension)
ITEM_NAME	VARCHAR2(255)	Physical name of the item
CUBE_ID	NUMBER(9)	ID of this cube
CUBE_NAME	VARCHAR2(255)	Physical name of this cube
DIMENSION_ID	NUMBER(9)	ID of the associated dimension
DIMENSION_NAME	VARCHAR2(255)	Physical name of the associated name
DIMENSION_ALIAS	VARCHAR2(255)	Alias of the associated dimension (name of the foreign key in the cube)
COLUMN_ID	NUMBER(9)	ID of the implementing column for the item
COLUMN_NAME	VARCHAR2(255)	Physical name of the implementing column for the item
POSITION	NUMBER	Position of the implementing column
TABLE_ID	NUMBER(9)	ID of the implementing table for this cube
TABLE_NAME	VARCHAR2(255)	Physical name of the implementing table

Table 2–67 (Cont.) ALL_IV_CUBE_IMPLS

Column Name	Data Type	Description
FOREIGN_KEY_ID	NUMBER(9)	ID of the foreign key pointing to the dimension
FOREIGN_KEY_NAME	VARCHAR2(255)	Physical name of the foreign key
DIM_IMPLEMENTATION_ID	NUMBER(9)	Not used (Value is set to NULL)

Table 2–68 ALL_IV_DIM_IMPLS

Column Name	Data Type	Description
IMPLEMENTATION_ID	NUMBER(9)	ID of the table in the case of STAR implementtion (this column will be updated in future)
LEVEL_ID	NUMBER(9)	ID of the level
DIMENSION_ID	NUMBER(9)	ID of the dimension
DIMENSION_NAME	VARCHAR2(255)	Physical name of the dimension
LEVEL_NAME	VARCHAR2(255)	Name of the level
TABLE_NAME	VARCHAR2(255)	Name of the table

Table 2–69 ALL_IV_DIM_LEVEL_IMPLS

Column Name	Data Type	Description
IMPLEMENTATION_ID	NUMBER(9)	ID of the level attribute (this column will be updated in future)
ITEM_ID	NUMBER(9)	ID of the item belonging to this level
ITEM_TYPE	VARCHAR2(18)	Type of the item (constant value: Level Attribute)
ITEM_NAME	VARCHAR2(255)	Physical name of the item
DIMENSION_ID	NUMBER(9)	ID of the dimension
DIMENSION_NAME	VARCHAR2(255)	Physical name of the dimension
LEVEL_ID	NUMBER(9)	ID of this level
LEVEL_NAME	VARCHAR2(255)	Physical name of this level
COLUMN_ID	NUMBER(9)	ID of the implementation column
COLUMN_NAME	VARCHAR2(255)	Physical name of the implementation column
TABLE_ID	NUMBER(9)	ID of the implementation table for this level
TABLE_NAME	VARCHAR2(255)	Physical name of the implementation table for this level

Table 2–70 ALL_IV_NESTED_TABLES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
NESTED_TABLE_ID	NUMBER(9)	ID of the nested table
NESTED_TABLE_NAME	VARCHAR2(255)	Name of the nested table
BUSINESS_NAME	VARCHAR2(1000)	Business name of the nested table
DESCRIPTION	VARCHAR2(4000)	Description of the nested table

Table 2–70 (Cont.) ALL_IV_NESTED_TABLES

Column Name	Data Type	Description
IS_VALID	VARCHAR2(13)	Is the nested table valid
BASE_ELEMENT_NAME	VARCHAR2(767)	Name of the base element
BASE_ELEMENT_PRECISION	NUMBER	Precision of the base element
BASE_ELEMENT_SCALE	NUMBER	Scale of the base element
BASE_ELEMENT_LENGTH	NUMBER	Length of the base element
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.4 Flat Files Views

Table 2–71 ALL_IV_FIELDS

Column Name	Data Type	Description
RECORD_ID	NUMBER(9)	ID of the record this field belongs to
RECORD_NAME	VARCHAR2(255)	Physical name of the record
FIELD_ID	NUMBER(9)	ID of this field
FIELD_NAME	VARCHAR2(255)	Physical name of this field
BUSINESS_NAME	VARCHAR2(4000)	Business name of this field
DESCRIPTION	VARCHAR2(4000)	Description of this field
POSITION	NUMBER(9)	Position of this field
DATA_TYPE	VARCHAR2(255)	Data type of this field
LENGTH	NUMBER(9)	Data length of this field
PRECISION	NUMBER(9)	Data precision of this field
SCALE	NUMBER(9)	Data scale of this field
PICTURE	VARCHAR2(40)	Picture of the field
SIGN_TYPE	NUMBER(9)	Sign type of the field
USAGE	VARCHAR2(40)	Usage of the field
MASK	VARCHAR2(255)	Mask of the field
NULLIF	VARCHAR2(40)	Nullif value of the field
DEFAULTIF	VARCHAR2(40)	Defaultif value of the field
SQL_DATA_TYPE	VARCHAR2(40)	SQL data type of the field
SQL_LENGTH	NUMBER(9)	SQL data length of the field
SQL_PRECISION	NUMBER(9)	SQL precision of the field
SQL_SCALE	NUMBER(9)	SQL data scale of the field
START_POSITION	VARCHAR2(40)	Start position of the field
OCCURS	NUMBER(9)	Occurs of the attribute array within the structure (identified by next column)

Table 2–71 (Cont.) ALL_IV_FIELDS

Column Name	Data Type	Description
STRUCTURE_ID	NUMBER(9)	ID of the structure containing field
STRUCTURE_NAME	VARCHAR2(255)	Physical name of the structure
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–72 ALL_IV_FILES

Column Name	Data Type	Description
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the module this file belongs to
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Physical name of the module
FILE_ID	NUMBER(9)	ID of this file
FILE_NAME	VARCHAR2(255)	Physical name of this file
BUSINESS_NAME	VARCHAR2(4000)	Business name of this file
DESCRIPTION	VARCHAR2(4000)	Description of the file
FILE_FORMAT	VARCHAR2(10)	Format of this file
IS_VALID	VARCHAR2(13)	Is this file valid
RECORD_CLASSIFIER_POSITION	NUMBER(9)	Record classifier position of this file
RECORD_CLASSIFIER_LENGTH	NUMBER(9)	Record classifier length of this file
RECORD_SIZE	VARCHAR2(40)	Record size of this file
N_PHYSICAL_RECORDS_IN_LOGICAL	NUMBER(9)	Number of physical records for each logical record
CONTINUATION_AT_END	CHAR(1)	Continuation at end or not
CONTINUATION_DELIMITER	VARCHAR2(40)	Continuation delimiter symbol
RECORD_DELIMITER	VARCHAR2(40)	Record delimiter symbol
FIELD_DELIMITER	VARCHAR2(40)	Field delimiter symbol
TEXT_START_DELIMITER	VARCHAR2(1)	Text start delimiter symbol
TEXT_END_DELIMITER	VARCHAR2(1)	Text end delimiter symbol
SOURCE_FROM	VARCHAR2(4000)	Directory path of this file
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–73 ALL_IV_RECORDS

Column Name	Data Type	Description
FILE_ID	NUMBER(9)	ID of the file this record belongs to
FILE_NAME	VARCHAR2(255)	Physical name of the file
RECORD_ID	NUMBER(9)	ID of this record
RECORD_NAME	VARCHAR2(255)	Physical name of this record
BUSINESS_NAME	VARCHAR2(4000)	Business name of this record
DESCRIPTION	VARCHAR2(4000)	Description of this record
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

2.5 Collections Views

Note: The view ALL_IV_COLLECTIONS replaces the CLASSIFICATION view.

Table 2–74 ALL_IV_COLLECTIONS

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project this collection belongs to (this view replaces old classification view)
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
COLLECTION_ID	NUMBER(9)	ID of this collection
COLLECTION_NAME	VARCHAR2(255)	Physical name of this collection
BUSINESS_NAME	VARCHAR2(4000)	Business name of this collection
DESCRIPTION	VARCHAR2(4000)	Description of this collection
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–75 ALL_IV_COLLECTION_REFERENCES

Column Name	Data Type	Description
COLLECTION_ID	NUMBER(9)	ID of the collection this reference belongs to (this view replaces old classification_item view)
COLLECTION_NAME	VARCHAR2(255)	Physical name of the collection
COLLECTION_REFERENCE_ID	NUMBER(9)	ID of this collection reference
COLLECTION_REFERENCE_TYPE	VARCHAR2(4000)	Type of this collection reference
COLLECTION_REFERENCE_NAME	VARCHAR2(255)	Physical name of the collection reference

Table 2–75 (Cont.) ALL_IV_COLLECTION_REFERENCES

Column Name	Data Type	Description
BUSINESS_NAME	VARCHAR2(4000)	Business name of this collection reference
DESCRIPTION	VARCHAR2(4000)	Description of this collection reference
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

2.6 Function Model Views

Table 2–76 ALL_IV_FUNCTIONS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this function belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
FUNCTION_LIBRARY_ID	NUMBER(9)	ID of the function library this function belongs to
FUNCTION_LIBRARY_NAME	VARCHAR2(255)	Physical name of the function library
FUNCTION_ID	NUMBER(9)	ID of this function
FUNCTION_NAME	VARCHAR2(255)	Physical name of this function
FUNCTION_TYPE	VARCHAR2(13)	Type of this function (function, procedure, table function)
SIGNATURE	VARCHAR2(4000)	Signature of this function
IS_VALID	VARCHAR2(13)	Is this function valid
BUSINESS_NAME	VARCHAR2(4000)	Business name of this function
DESCRIPTION	VARCHAR2(4000)	Description of this function
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–77 ALL_IV_FUNCTION_LIBRARIES

Column Name	Data Type	Description
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the module this function library belongs to
INFORMATION_SYSTEM_ NAME	VARCHAR2(255)	Physical name of the module
FUNCTION_LIBRARY_ID	NUMBER(9)	ID of this function library
FUNCTION_LIBRARY_NAME	VARCHAR2(255)	Physical name of this function library
FUNCTION_LIBRARY_TYPE	VARCHAR2(40)	Type of this function library
IS_VALID	VARCHAR2(13)	Is this function library valid
BUSINESS_NAME	VARCHAR2(4000)	Business name of this function library

Table 2–77 (Cont.) ALL_IV_FUNCTION_LIBRARIES

Column Name	Data Type	Description
DESCRIPTION	VARCHAR2(4000)	Description of this function library
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–78 ALL_IV_FUNCTION_PARAMETERS

Column Name	Data Type	Description
FUNCTION_ID	NUMBER(9)	ID of the function this parameter belongs to
FUNCTION_NAME	VARCHAR2(255)	Physical name of the function
PARAMETER_ID	NUMBER(9)	ID of this parameter
PARAMETER_NAME	VARCHAR2(255)	Physical name of this parameter
PARAMETER_TYPE	VARCHAR2(40)	Type of this parameter
BUSINESS_NAME	VARCHAR2(4000)	Business name of this parameter
DESCRIPTION	VARCHAR2(4000)	Description of this parameter
POSITION	NUMBER(9)	Position of this parameter within the function
DATA_TYPE	VARCHAR2(255)	Data type of this parameter
LENGTH	NUMBER(9)	Data length of this parameter
PRECISION	NUMBER(9)	Data precision of this parameter
SCALE	NUMBER(9)	Data scale of this parameter
DEFAULT_VALUE	VARCHAR2(4000)	Default value of this parameter
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–79 ALL_IV_TABLE_FUNCTIONS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this table function belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
FUNCTION_LIBRARY_ID	NUMBER(9)	ID of the function library this table function belongs to
FUNCTION_LIBRARY_ NAME	VARCHAR2(255)	Physical name of the function library
FUNCTION_ID	NUMBER(9)	ID of this table function
FUNCTION_NAME	VARCHAR2(255)	Physical name of this table function
FUNCTION_TYPE	VARCHAR2(13)	Type of this table function (constant value: Table Function)
SIGNATURE	VARCHAR2(4000)	Signature of this table function

Table 2–79 (Cont.) ALL_IV_TABLE_FUNCTIONS

Column Name	Data Type	Description
IS_VALID	VARCHAR2(13)	Is this table function valid
BUSINESS_NAME	VARCHAR2(4000)	Business name of this table function
DESCRIPTION	VARCHAR2(4000)	Description of this table function
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–80 ALL_IV_FUNCTION_IMPLS

Column Name	Data Type	Description
FUNCTION_ID	NUMBER(9)	ID of the function
FUNCTION_NAME	VARCHAR2(255)	Physical name of the function
FUNCTION_IMPLEMENTATION_ID	NUMBER(9)	ID of the function implementation for this function
FUNCTION_IMPLEMENTATION_NAME	VARCHAR2(255)	Physical name of the function implementation
LANGUAGE	VARCHAR2(255)	Name of the language being used in the implementation
SCRIPT	VARCHAR2(4000)	Implementation script for this function
BUSINESS_NAME	VARCHAR2(4000)	Business name for this implementation
DESCRIPTION	VARCHAR2(4000)	Description for this implementation
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

2.7 Configuration Model Views

Table 2–81 ALL_IV_OBJECT_CONFIGURATIONS

Column Name	Data Type	Description
CONFIGURED_OBJECT_ID	NUMBER(9)	ID of the object being configured
CONFIGURED_OBJECT_NAME	VARCHAR2(255)	Physical name of the object
CONFIGURED_OBJECT_TYPE	VARCHAR2(4000)	Type of the object
CONFIGURATION_PARAMETER_KEY	VARCHAR2(128)	Key of configuration parameter
CONFIGURATION_PARAMETER_ NAME	VARCHAR2(64)	Name of configuration parameter
PARAMETER_NLSKEY	VARCHAR2(64)	National Language Support (NLS) key of the parameter
CONFIGURATION_PARAMETER_TYPE	CHAR(23)	Type of the configuration parameter
ARGUMENT	VARCHAR2(128)	Value of the configuration parameter

Table 2–81 (Cont.) ALL_IV_OBJECT_CONFIGURATIONS

Column Name	Data Type	Description
GROUP_NAME	VARCHAR2(322)	Name of the configuration group
GROUP_NLSKEY	VARCHAR2(64)	National Language Support (NLS) key of the configuration group
LANGUAGE	VARCHAR2(64)	Name of the language being used for this configuration

Table 2–82 ALL_IV_CONFIGURATIONS

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
CONFIGURATION_ID	NUMBER(9)	ID of the configuration
CONFIGURATION_NAME	VARCHAR2(255)	Name of the configuration
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–83 ALL_IV_CONFIG_TEMPLATES

Column Name	Datatype	Description
CONFIG_TEMPLATE_ID	NUMBER (9)	ID of the configuration template
CONFIG_TEMPLATE_NAME	VARCHAR2 (1000)	Name of the configuration template
BUSINESS_NAME	VARCHAR2(1000)	Business name of the configuration template
DESCRIPTION	VARCHAR2(4000)	Description of the configuration template
OVERRIDE_MODEL	VARCHAR2(40)	Override model
OVERRIDE_MODEL_ID	NUMBER(9)	Override model ID
CONFIG_TEMPLATE_SET	VARCHAR2(1000)	Config template set
CONFIG_TEMPLATE_SET_ID	NUMBER(9)	Config template set ID
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–84 ALL_IV_CONFIG_TEMPLATE_SETS

Column Name	Datatype	Description
CONFIG_TEMPLATE_SET_ID	NUMBER (9)	ID of the configuration template set
CONFIG_TEMPLATE_SET_ NAME	VARCHAR2 (1000)	Name of the configuration template set
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the configuration template set

Table 2–84 (Cont.) ALL_IV_CONFIG_TEMPLATE_SETS

Column Name	Datatype	Description
DESCRIPTION	VARCHAR2 (4000)	Description
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–85 ALL_IV_CONTROL_CENTERS

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
CONTROL_CENTER_ID	NUMBER(9)	ID of the control center
CONTROL_CENTER_NAME	VARCHAR2(255)	Name of the control center
BUSINESS_NAME	VARCHAR2(1000)	Business name of the control center
DESCRIPTION	VARCHAR2(4000)	Description of the control center
HOST	VARCHAR2(255)	Host of the control center
SERVICE_NAME	VARCHAR2(4000)	Service name of the control center
PORT	NUMBER	Port of the control center
USERNAME	VARCHAR2(4000)	User name who will connect to the control center
SCHEMA	VARCHAR2(40)	Schema for the control center
IS_VALID	VARCHAR2(13)	The result of the last validation performed against the control center
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.8 Deployment Model Views

Table 2–86 ALL_IV_CONNECTORS

Column Name	Data Type	Description
LOCATION_ID	NUMBER(9)	ID of the location owning this connector
LOCATION_NAME	VARCHAR2(255)	Physical name of the location
CONNECTOR_ID	NUMBER(9)	ID of this connector
CONNECTOR_NAME	VARCHAR2(255)	Physical name of this connector
BUSINESS_NAME	VARCHAR2(4000)	Business name of this connector
DESCRIPTION	VARCHAR2(4000)	Description of this connector
REFERENCED_LOCATION_ID	NUMBER(9)	ID of the location this connector references to

Table 2–86 (Cont.) ALL_IV_CONNECTORS

Column Name	Data Type	Description
REFERENCED_LOCATION_NAME	VARCHAR2(255)	Physical name of the location this connector references to
IS_VALID	VARCHAR2(13)	Is this connector valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–87 ALL_IV_LOCATIONS

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project this location belongs to
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
LOCATION_ID	NUMBER(9)	ID of this location
LOCATION_NAME	VARCHAR2(255)	Physical name of this location
LOCATION_TARGET_TYPE	VARCHAR2(40)	Target type of this location
LOCATION_TARGET_VERSION	VARCHAR2(40)	Target version of this location
APPLICATION_TYPE	VARCHAR2(255)	Application type of the location connected to
SYSTEM_TYPE	VARCHAR2(255)	System type of this location connected to
IS_VALID	VARCHAR2(13)	Is this location valid
BUSINESS_NAME	VARCHAR2(4000)	Business name of this location
DESCRIPTION	VARCHAR2(4000)	Description of this location
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–88 ALL_IV_RUNTIME_REPOSITORIES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project this repository belongs to (also called runtime location, or simply, location)
PROJECT_NAME	VARCHAR2(255)	Physical name of the project
LOCATION_ID	NUMBER(9)	ID of this runtime location
LOCATION_NAME	VARCHAR2(255)	Physical name of this runtime location
LOCATION_TYPE	VARCHAR2(255)	Type of this runtime location
APPLICATION_TYPE	VARCHAR2(255)	Type of the application this location connected to
SYSTEM_TYPE	VARCHAR2(255)	Type of the system this location connected to
BUSINESS_NAME	VARCHAR2(4000)	Business name of this runtime location
DESCRIPTION	VARCHAR2(4000)	Description of this runtime location

Table 2–88 (Cont.) ALL_IV_RUNTIME_REPOSITORIES

Column Name	Data Type	Description
HOST	VARCHAR2(40)	Host name of the connection for this location
SERVICE_NAME	VARCHAR2(40)	Service name of the connection for this location
PORT	NUMBER(9)	Port of the connection for this location
USERNAME	VARCHAR2(40)	User name of the connection for this location
SCHEMA	VARCHAR2(40)	Schema name of the connection for this location
IS_VALID	VARCHAR2(13)	Is this runtime location valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

2.9 Mapping Model Views

Table 2–89 ALL_IV_XFORM_MAPS

Column Name	Data Type	Description
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the module this map belongs to
INFORMATION_SYSTEM_NAME	VARCHAR2(255)	Physical name of the module
MAP_ID	NUMBER(9)	ID of this map
MAP_NAME	VARCHAR2(255)	Physical name of this map
BUSINESS_NAME	VARCHAR2(4000)	Business name of this map
DESCRIPTION	VARCHAR2(4000)	Description of this map
COMPOSITE_MAP_COMPONENT_ID	NUMBER(9)	Same as MAP_ID
COMPOSITE_MAP_COMPONENT_NAME	VARCHAR2(255)	Same as MAP_NAME
IS_VALID	VARCHAR2(13)	Is this map valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–90 ALL_IV_XFORM_MAP_COMPONENTS

Column Name	Data Type	Description
MAP_ID	NUMBER(9)	ID of the map this map component belongs to
MAP_NAME	VARCHAR2(255)	Physical name of the map
MAP_COMPONENT_ID	NUMBER(9)	ID of this map component (also called map operator)
MAP_COMPONENT_NAME	VARCHAR2(255)	Physical name of this map component
BUSINESS_NAME	VARCHAR2(4000)	Business name of this map component
DESCRIPTION	VARCHAR2(4000)	Description of this map component

Table 2–90 (Cont.) ALL_IV_XFORM_MAP_COMPONENTS

Column Name	Data Type	Description
OPERATOR_TYPE	VARCHAR2(4000)	Type of this map component (for example, Filter, Joiner, Table)
COMPOSITE_MAP_COMPONENT_ID	NUMBER(9)	Same as MAP_COMPONENT_ID
COMPOSITE_MAP_COMPONENT_NAME	VARCHAR2(255)	Same as MAP_COMPONENT_NAME
DATA_ENTITY_ID	NUMBER(9)	ID of the data entity this map component synchronized to
DATA_ENTITY_NAME	VARCHAR2(255)	Physical name of the data entity
DATA_ENTITY_TYPE	VARCHAR2(4000)	Type of the data entity
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–91 ALL_IV_XFORM_MAP_PARAMETERS

Column Name	Data Type	Description
MAP_COMPONENT_ID	NUMBER(9)	ID of the map component this parameter belongs to
MAP_COMPONENT_NAME	VARCHAR2(255)	Physical name of the map component
PARAMETER_ID	NUMBER(9)	ID of this parameter
PARAMETER_NAME	VARCHAR2(255)	Physical name of this parameter
BUSINESS_NAME	VARCHAR2(4000)	Business name of this parameter
DESCRIPTION	VARCHAR2(4000)	Description of this parameter
MAP_ID	NUMBER(9)	ID of the map containing the map component
MAP_NAME	VARCHAR2(255)	Physical name of the map
PARAMETER_GROUP_NAME	VARCHAR2(255)	Physical name of the parameter group name
PARAMETER_GROUP_ID	NUMBER(9)	ID of the parameter group
PARAMETER_TYPE	VARCHAR2(5)	Type of the parameter (IN, OUT, INOUT)
POSITION	NUMBER(9)	Position of the parameter within the group
DATA_TYPE	VARCHAR2(40)	Data type of the parameter
TRANSFORMATION_EXPRESSION	VARCHAR2(4000)	Textual expression of the transformation for this parameter
DATA_ITEM_ID	NUMBER(9)	ID of the data item this parameter synchronized to
DATA_ITEM_TYPE	VARCHAR2(40)	Type of the data item
DATA_ITEM_NAME	VARCHAR2(255)	Physical name of the data item
SOURCE_PARAMETER_ID	NUMBER(9)	ID of the source parameter (where this parameter connected from)
SOURCE_PARAMETER_NAME	VARCHAR2(255)	Physical name of the source parameter
UPDATED_ON	DATE	Update timestamp

Table 2–91 (Cont.) ALL_IV_XFORM_MAP_PARAMETERS

Column Name	Data Type	Description
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–92 ALL_IV_XFORM_MAP_PROPERTIES

Column Name	Data Type	Description
MAP_COMPONENT_ID	NUMBER(9)	ID of the map component this property belongs to
MAP_COMPONENT_NAME	VARCHAR2(255)	Physical name of the map component
PROPERTY_ID	NUMBER(9)	ID of this property
PROPERTY_NAME	VARCHAR2(255)	Physical name of this property
BUSINESS_NAME	VARCHAR2(4000)	Business name of this property
DESCRIPTION	VARCHAR2(4000)	Description of this property
PROPERTY_GROUP_NAME	VARCHAR2(255)	Physical name of this property group
PROPERTY_VALUE	VARCHAR2(4000)	Value of this property

Table 2–93 ALL_IV_XFORM_MAP_DETAILS

Column Name	Data Type	Description
MAP_COMPONENT_ID	NUMBER(9)	ID of the map component
MAP_COMPONENT_NAME	VARCHAR2(255)	Name of the map component
PARAMETER_NAME	VARCHAR2(255)	Name of the parameter
PARAMETER_ID	NUMBER(9)	ID of the parameter
POSITION	NUMBER(9)	Position
BUSINESS_NAME	VARCHAR2(1000)	Business name of map component
TRANSFORMATION_EXPRESSION	VARCHAR2(4000)	Transformation expression
DESCRIPTION	VARCHAR2(4000)	Description of map component
SOURCE_EXPRESSION	VARCHAR2(4000)	Source expression

Table 2–94 ALL_IV_PLUGGABLE_MAPS

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
MAP_LIBRARY_ID	NUMBER	ID of the map library
MAP_LIBRARY_NAME	VARCHAR2(255)	Name of the map library
MAP_ID	NUMBER(9)	ID of the map
MAP_NAME	VARCHAR2(255)	Name of the map
BUSINESS_NAME	VARCHAR2(1000)	Business name of the map
DESCRIPTION	VARCHAR2(4000)	Description of the map

Table 2–94 (Cont.) ALL_IV_PLUGGABLE_MAPS

Column Name	Data Type	Description
IS_VALID	VARCHAR2(13)	Is the map valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–95 ALL_IV_PLUGGABLE_MAP_LIBRARIES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
MAP_LIBRARY_ID	NUMBER(9)	ID of the map library
MAP_LIBRARY_NAME	VARCHAR2(255)	Name of the map library
BUSINESS_NAME	VARCHAR2(1000)	Business name of the map library
DESCRIPTION	VARCHAR2(4000)	Description of the map library
IS_VALID	VARCHAR2(13)	Is the library valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–96 ALL_IV_PLUG_MAP_PARAMETERS

Column Name	Data Type	Description
MAP_COMPONENT_ID	NUMBER(9)	ID of the map component
MAP_COMPONENT_NAME	VARCHAR2(255)	Name of the map component
PARAMETER_ID	NUMBER(9)	ID of the parameter
PARAMETER_NAME	VARCHAR2(255)	Name of the parameter
BUSINESS_NAME	VARCHAR2(1000)	Business name of the parameter
DESCRIPTION	VARCHAR2(4000)	Description of the parameter
MAP_ID	NUMBER(9)	ID of the map
MAP_NAME	VARCHAR2(255)	Name of the map
PARAMETER_GROUP_NAME	VARCHAR2(255)	Name of the parameter group
PARAMETER_GROUP_ID	NUMBER(9)	ID of the parameter group
PARAMETER_TYPE	VARCHAR2(5)	Type of the parameter
POSITION	NUMBER(9)	The position of the parameter
DATA_TYPE	VARCHAR2(4000)	The data type of the parameter
TRANSFORMATION_EXPRESSION	VARCHAR2(4000)	The expression of the parameter
DATA_ITEM_ID	NUMBER(9)	ID of the data item

Table 2–96 (Cont.) ALL_IV_PLUG_MAP_PARAMETERS

Column Name	Data Type	Description
DATA_ITEM_TYPE	VARCHAR2	Type of the data item
DATA_ITEM_NAME	VARCHAR2(255)	Name of the data item
SOURCE_PARAMETER_ID	NUMBER(9)	The ID of the parameter where this parameter is connected from
SOURCE_PARAMETER_NAME	VARCHAR2(255)	The name of the parameter where this parameter is connected from
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–97 ALL_IV_PLUG_MAP_COMPONENTS

Column Name	Data Type	Description
MAP_ID	NUMBER(9)	ID of the map
MAP_NAME	VARCHAR2(255)	Name of the map
MAP_COMPONENT_ID	NUMBER(9)	ID of the map component
MAP_COMPONENT_NAME	VARCHAR2(255)	Name of the map component
BUSINESS_NAME	VARCHAR2(1000)	Business name of the map component
DESCRIPTION	VARCHAR2(4000)	Description of the map component
OPERATOR_TYPE	VARCHAR2(4000)	The operator type of the map component
COMPOSITE_COMPONENT_ID	NUMBER(9)	ID of the map component
COMPOSITE_COMPONENT_NAME	VARCHAR2(255)	Name of the map component
DATA_ENTITY_ID	NUMBER(9)	ID of the data entity
DATA_ENTITY_TYPE	VARCHAR2(4000)	Type of data entity
DATA_ENTITY_NAME	VARCHAR2(255)	Name of the data entity
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.10 Process Flow Model Views

Table 2–98 ALL_IV_PACKAGES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the module this process package belongs to
SCHEMA_NAME	VARCHAR2(255)	Physical name of the module
PACKAGE_ID	NUMBER(9)	ID of this process package
PACKAGE_NAME	VARCHAR2(255)	Physical name of this process package

Table 2–98 (Cont.) ALL_IV_PACKAGES

Column Name	Data Type	Description
BUSINESS_NAME	VARCHAR2(4000)	Business name of this process package
DESCRIPTION	VARCHAR2(4000)	Description of this process package
IS_VALID	VARCHAR2(13)	Is this process package valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–99 ALL_IV_PROCESSES

Column Name	Data Type	Description
PACKAGE_ID	NUMBER(9)	ID of the process package this process belongs to
PACKAGE_NAME	VARCHAR2(255)	Physical name of the process package
PARENT_PROCESS_ID	NUMBER(9)	ID of the parent process for this process
PARENT_PROCESS_NAME	VARCHAR2(255)	Physical name of the parent process for this process
PROCESS_ID	NUMBER(9)	ID of this process
PROCESS_NAME	VARCHAR2(255)	Physical name of this process
BUSINESS_NAME	VARCHAR2(4000)	Business name of this process
DESCRIPTION	VARCHAR2(4000)	Description of this process
BOUND_OBJECT_ID	NUMBER(9)	ID of the bound object
BOUND_OBJECT_NAME	VARCHAR2(255)	Name of the bound object
IS_VALID	VARCHAR2(13)	Is this process valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–100 ALL_IV_PROCESS_ACTIVITIES

Column Name	Data Type	Description
PROCESS_ID	NUMBER(9)	ID of the process this activity belongs to
PROCESS_NAME	VARCHAR2(255)	Physical name of the process
ACTIVITY_ID	NUMBER(9)	ID of this process activity
ACTIVITY_NAME	VARCHAR2(255)	Physical name of this activity
ACTIVITY_TYPE	VARCHAR2(4000)	Type of this activity
BOUND_OBJECT_ID	NUMBER(9)	ID of the bound object
BOUND_OBJECT_NAME	VARCHAR2(255)	Name of the bound object
UPDATED_ON	DATE	Update timestamp

Table 2–100 (Cont.) ALL_IV_PROCESS_ACTIVITIES

Column Name	Data Type	Description
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–101 ALL_IV_PROCESS_PARAMETERS

Column Name	Data Type	Description
PARAMETER_OWNER_ID	NUMBER(9)	ID of the owning object for this parameter
PARAMETER_OWNER_NAME	VARCHAR2(255)	Physical name of the owning object for this parameter
PARAMETER_OWNER_TYPE	CHAR(14)	Type of the owning object
PARAMETER_ID	NUMBER(9)	ID of this parameter
PARAMETER_NAME	VARCHAR2(255)	Physical name of this parameter
BUSINESS_NAME	VARCHAR2(4000)	Business name of this parameter
DESCRIPTION	VARCHAR2(4000)	Description of this parameter
POSITION	NUMBER(9)	Position of this parameter
DATA_TYPE	VARCHAR2(40)	Data type of this parameter
DEFAULT_VALUE	VARCHAR2(4000)	Default value for this parameter
DIRECTION	VARCHAR2(3)	Direction of this parameter (IN, OUT)
IS_FINAL	CHAR(1)	Is process final
BOUNDDATA_ID	NUMBER(9)	ID of the bound data for this parameter
BOUNDDATA_NAME	VARCHAR2(255)	Physical name of the bound data
BOUNDDATA_TYPE	VARCHAR2(40)	Type of the bound data
BOUNDDATA_VALUE	VARCHAR2(4000)	Value of the bound data
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–102 ALL_IV_PROCESS_TRANSITIONS

Column Name	Data Type	Description
PROCESS_ID	NUMBER(9)	ID of the process this transition belongs to
PROCESS_NAME	VARCHAR2(255)	Physical name of the process
TRANSITION_ID	NUMBER(9)	ID of this transition
TRANSITION_NAME	VARCHAR2(255)	Physical name of this transition
BUSINESS_NAME	VARCHAR2(4000)	Business name of this transition
DESCRIPTION	VARCHAR2(4000)	Description of this transition
CONDITION	VARCHAR2(255)	Condition of this transition

Table 2–102 (Cont.) ALL_IV_PROCESS_TRANSITIONS

Column Name	Data Type	Description
TRANSITION_ORDER	NUMBER(9)	Order of this transition
SOURCE_ACTIVITY_ID	NUMBER(9)	ID of the source activity for this transition
SOURCE_ACTIVITY_NAME	VARCHAR2(255)	Physical name of the source activity
TARGET_ACTIVITY_ID	NUMBER(9)	ID of the target activity for this transition
TARGET_ACTIVITY_NAME	VARCHAR2(255)	Physical name of the target activity
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–103 ALL_IV_PROCESS_VARIABLES

Column Name	Data Type	Description
PROCESS_ID	NUMBER(9)	ID of the process this variable belongs to
PROCESS_NAME	VARCHAR2(255)	Physical name of the process
VARIABLE_ID	NUMBER(9)	ID of this process variable
VARIABLE_NAME	VARCHAR2(255)	Physical name of this process variable
BUSINESS_NAME	VARCHAR2(4000)	Business name of this process variable
DESCRIPTION	VARCHAR2(4000)	Description of this process variable
POSITION	NUMBER(9)	Position of this variable
DATA_TYPE	VARCHAR2(40)	Data type of this variable
DEFAULT_VALUE	VARCHAR2(4000)	Default value of this variable
IS_FINAL	CHAR(1)	Is process final
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(255)	Updated by user
CREATED_BY	VARCHAR2(255)	Created by user

Table 2–104 ALL_IV_SUB_PROCESSES

Column Name	Data Type	Description
PACKAGE_ID	NUMBER(9)	ID of the package
PACKAGE_NAME	VARCHAR2(255)	Name of the package
PARENT_PROCESS_ID	NUMBER(9)	ID of the parent process
PARENT_PROCESS_NAME	VARCHAR2(255)	Name of the parent process
PROCESS_ID	NUMBER(9)	ID of the process
PROCESS_NAME	VARCHAR2(255)	Name of the process
BUSINESS_NAME	VARCHAR2(1000)	Business name of the condition
DESCRIPTION	VARCHAR2(4000)	Description of the condition

Table 2–104 (Cont.) ALL_IV_SUB_PROCESSES

Column Name	Data Type	Description
BOUND_OBJECT_ID	NUMBER(9)	ID of the bound object
BOUND_OBJECT_NAME	VARCHAR2(40)	Name of the bound object
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.11 Data Profiling Views

Table 2–105 ALL_IV_PROFILES

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
PROFILE_ID	NUMBER(9)	ID of the profile
PROFILE_NAME	VARCHAR2(255)	Name of the profile
BUSINESS_NAME	VARCHAR2(1000)	Business name of the profile
DESCRIPTION	VARCHAR2(4000)	Description of the profile
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–106 ALL_IV_PROFILE_COLUMNS

Column Name	Data Type	Description
PROFILE_ID	NUMBER(9)	ID of the profile
PROFILE_NAME	VARCHAR2(255)	Name of the profile
PROFILE_OBJECT_ID	NUMBER(9)	ID of the profile object
ENTITY_ID	NUMBER(9)	ID of the entity
ENTITY_NAME	VARCHAR2(255)	Name of the entity
PROFILE_COLUMN_ID	NUMBER(9)	ID of the profile column
COLUMN_ID	NUMBER(9)	ID of the column
COLUMN_NAME	VARCHAR2(255)	Name of the column
BUSINESS_NAME	VARCHAR2(1000)	Business name of column
AVG_VALUE	VARCHAR2(40)	Average value of column if column has a numeric data type
MAX_VALUE	VARCHAR2(4000)	Maximum value stored in the column
MIN_VALUE	VARCHAR2(4000)	Minimum value stored in the column

Table 2–106 (Cont.) ALL_IV_PROFILE_COLUMNS

Column Name	Data Type	Description
MEDIAN_VALUE	VARCHAR2(4000)	Median value of column if column has a numeric data type
STDDEV_VALUE	VARCHAR2(40)	Standard deviation of column if column has a numeric data type
NUM_NULLS	NUMBER	Number of null values stored in the column
NUM_DISTINCT	NUMBER	Number of distinct values stored in the column
CONSENSUS_DATATYPE	VARCHAR2(40)	The discovered data type for the column
CONSENSUS_DATATYPE_CNT	VARCHAR2(40)	The number of rows that have the consensus data type
CONSENSUS_LENGTH	NUMBER	The predominant length of the column
CONSENSUS_LENGTH_CNT	NUMBER	The number of rows with the predominant length
CONSENSUS_PRECISION	NUMBER	The predominant precision of the column
CONSENSUS_PRECISION_CNT	NUMBER	The number of rows with the predominant precision
CONSENSUS_SCALE	NUMBER	The predominant scale of the column
CONSENSUS_SCALE_CNT	NUMBER	The number of rows with the predominant scale
COMMON_FORMAT	VARCHAR2(40)	The discovered common format
COMMON_FORMAT_CNT	VARCHAR2(40)	The number of rows that have this discovered common format
DOMINANT_CHARACTER_PATTERN	VARCHAR2(4000)	The discovered pattern at the character level of the column
DOMINANT_CHARACTER_PATTERN_CNT	VARCHAR2(40)	The number of rows that satisfy the character level pattern
DOMINANT_WORD_PATTERN	VARCHAR2(4000)	The discovered pattern at the word level of the column
DOMINANT_WORD_PATTERN_CNT	VARCHAR2(40)	The number of rows that satisfy the word level pattern
MAX_LENGTH	NUMBER	Maximum length of the values stored in the column
MIN_LENGTH	NUMBER	Minimum length of the values stored in the column
MAX_PRECISION	NUMBER	Maximum precision of the values stored in the column
MIN_PRECISION	NUMBER	Minimum precision of the values stored in the column
MAX_SCALE	NUMBER	Maximum scale of the values stored in the column
MIN_SCALE	NUMBER	Minimum scale of the values stored in the column
CFORMAT_COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that satisfy the discovered common format

Table 2–106 (Cont.) ALL_IV_PROFILE_COLUMNS

Column Name	Data Type	Description
CFORMAT_DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row satisfies the discovered common format and which does not
CFORMAT_NONCOMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy the discovered common format
DATATYPE_COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that satisfy the discovered data type
DATATYPE_DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row satisfies the discovered data type and which does not
DATATYPE_NONCOMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy the discovered data type
DOMAIN_COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that contain the discovered domain values
DOMAIN_DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row contains the discovered domain values and which does not
DOMAIN_NONCOMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not contain the discovered domain values
NULL_COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that are null based on the configured null value
NULL_DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row is null and which isn't based on the configured null value
NULL_NONCOMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy the discovered common format
CHAR_PATTERN_COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that satisfy the discovered character pattern
CHAR_PATTERN_DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row satisfies the discovered character pattern and which does not
CHAR_PATTERN_NONCOM_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy the discovered character pattern
WORD_PATTERN_COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that satisfy the discovered word pattern
WORD_PATTERN_DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row satisfies the discovered word pattern and which does not
WORD_PATTERN_NONCOM_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy the discovered word pattern
UPDATED_ON	DATE	Update timestamp

Table 2–106 (Cont.) ALL_IV_PROFILE_COLUMNS

Column Name	Data Type	Description
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–107 ALL_IV_PROFILE_DOMAIN_VALUES

Column Name	Data Type	Description
PROFILE_OBJECT_ID	NUMBER(9)	ID of the profile object
PROFILE_COLUMN_ID	NUMBER(9)	ID of the profile column
COLUMN_NAME	VARCHAR2(255)	Name of the column
VALUE	VARCHAR2(4000)	Domain value
VALUE_COUNT	VARCHAR2(4000)	Number of rows that have this domain value
COMPLIANT	VARCHAR2(1)	Whether this domain value satisfies the domain value configuration

Table 2–108 ALL_IV_FUNCTIONAL_DEPENDENCIES

Column Name	Data Type	Description
PROFILE_ID	NUMBER(9)	ID of the profile
PROFILE_NAME	VARCHAR2(255)	Name of the profile
ENTITY_ID	NUMBER(9)	ID of the entity
ENTITY_NAME	VARCHAR2(255)	Name of the entity
FUNCTIONAL_DEPENDENCY_ID	NUMBER(9)	ID of the functional dependency
FUNCTIONAL_DEPENDENCY_NAME	VARCHAR2(255)	Name of the functional dependency
BUSINESS_NAME	VARCHAR2(1000)	Business name of the functional dependency
TYPE	VARCHAR2(15)	Type of functional dependency
DEPENDENT_COLUMN_ID	NUMBER(9)	ID of the dependent column in the functional dependency
FD_ERROR	NUMBER	The number of rows that do not satisfy this functional dependency
COMPLIANT_QUERY	VARCHAR2(4000)	The query that is used to retrieve the rows that satisfy this functional dependency
COMPLIANT_CNT_QUERY	VARCHAR2(4000)	Not used
NONCOMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy this functional dependency
NONCOMPLIANT_CNT_QUERY	VARCHAR2(4000)	Not used
DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row satisfies this functional dependency and which does not
UPDATED_ON	DATE	Update timestamp

Table 2–108 (Cont.) ALL_IV_FUNCTIONAL_DEPENDENCIES

Column Name	Data Type	Description
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–109 ALL_IV_PROFILE_FOREIGN_KEYS

Column Name	Data Type	Description
PROFILE_ID	NUMBER(9)	ID of the profile
PROFILE_NAME	VARCHAR2(255)	Name of the profile
ENTITY_ID	NUMBER(9)	ID of the entity
ENTITY_NAME	VARCHAR2(255)	Name of the entity
FOREIGN_KEY_ID	NUMBER(9)	ID of the row relationship
FOREIGN_KEY_NAME	VARCHAR2(255)	Name of the row relationship
BUSINESS_NAME	VARCHAR2(1000)	Business name of the foreign key
UNIQUE_KEY_ID	NUMBER(9)	ID of the unique key
IS_DISCOVERED	VARCHAR2(3)	Should a foreign key be created on the column
IS_DOCUMENTED	VARCHAR2(3)	Indicates if a foreign key exists in the data dictionary for the column
LOCAL_MAX_CARDINALITY	VARCHAR2(40)	Maximum number of values found on the local side
LOCAL_MIN_CARDINALITY	VARCHAR2(40)	Minimum number of values found on the local side
REMOTE_MAX_CARDINALITY	VARCHAR2(40)	Maximum number of values found on the remote side
REMOTE_MIN_CARDINALITY	VARCHAR2(40)	Minimum number of values found on the remote side
NUM_ORPHANS	VARCHAR2(40)	Number of distinct values found in the local column but not in the remote column
COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that satisfy the discovered row relationship
COMPLIANT_CNT_QUERY	VARCHAR2(4000)	Not used
NONCOMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy the discovered common format
NONCOMPLIANT_CNT_QUERY	VARCHAR2(4000)	Not used
DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row satisfies the discovered row relationship and which does not
CHILDLESS_QUERY	VARCHAR2(4000)	Not used
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–110 ALL_IV_PROFILE_KEY_COLUMN_USES

Column Name	Data Type	Description
KEY_ID	NUMBER(9)	ID of the key
KEY_TYPE	VARCHAR2(4000)	Type of the key
KEY_NAME	VARCHAR2(255)	Name of the key
COLUMN_ID	NUMBER(9)	ID of the column
COLUMN_NAME	VARCHAR2(255)	Physical name of the column
BUSINESS_NAME	VARCHAR2(1000)	Business name of the column
POSITION	NUMBER(9)	Position of the column in the set

Table 2–111 ALL_IV_PROFILE_OBJECTS

Column Name	Data Type	Description
PROFILE_ID	NUMBER(9)	ID of the profile
PROFILE_NAME	VARCHAR2(255)	Name of the profile
PROFILE_OBJECT_ID	NUMBER(9)	ID of the profile object
OBJECT_ID	NUMBER(9)	ID of the object
PROFILE_OBJECT_NAME	VARCHAR2(255)	Name of the profile object
OBJECT_NAME	VARCHAR2(255)	Physical name of the object
OBJECT_TYPE	VARCHAR2(4000)	Type of the object
BUSINESS_NAME	VARCHAR2(1000)	Business name of the object
DESCRIPTION	VARCHAR2(4000)	Description of the profile object
IS_VALID	VARCHAR2(13)	Is the profile object valid
ROW_COUNT	NUMBER	Number of rows in the object
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–112 ALL_IV_PROFILE_PATTERN_VALUES

Column Name	Data Type	Description
PROFILE_OBJECT_ID	NUMBER(9)	ID of the profile object
PROFILE_COLUMN_ID	NUMBER(9)	ID of the profile column
COLUMN_NAME	VARCHAR2(255)	Name of the column
VALUE	VARCHAR2(4000)	Pattern value
VALUE_COUNT	VARCHAR2(4000)	The number of rows that satisfy this discovered pattern value
TYPE	VARCHAR2(9)	0 for Word Pattern, 1 for Character Pattern

Table 2–112 (Cont.) ALL_IV_PROFILE_PATTERN_VALUES

Column Name	Data Type	Description
COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy this pattern value
NONCOMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy this pattern value
DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row satisfies this pattern value and which does not

Table 2–113 ALL_IV_PROFILE_RULES

Column Name	Data Type	Description
PROFILE_OBJECT_ID	NUMBER(9)	ID of the profile object
DATA_RULE_USAGE_ID	NUMBER(9)	ID of the data rule usage
DATA_RULE_USAGE_NAME	VARCHAR2(255)	Name of the data rule usage
COMPLIANT_ROW_COUNT	VARCHAR2(4000)	The number of rows that comply with the data rule

Table 2–114 ALL_IV_PROFILE_UNIQUE_KEYS

Column Name	Data Type	Description
PROFILE_ID	NUMBER(9)	ID of the profile
PROFILE_NAME	VARCHAR2(255)	Name of the profile
ENTITY_ID	NUMBER(9)	ID of the entity
ENTITY_NAME	VARCHAR2(255)	Name of the entity
UNIQUE_KEY_ID	NUMBER(9)	ID of the unique key
UNIQUE_KEY_NAME	VARCHAR2(255)	Name of the unique key
BUSINESS_NAME	VARCHAR2(1000)	Business name of the unique key
IS_DISCOVERED	VARCHAR2(3)	Should a unique key be created on the column
IS_DOCUMENTED	VARCHAR2(3)	Indicates if a unique key exists on the column in the data dictionary
UK_ERROR	VARCHAR2(40)	Number of rows that do not satisfy this unique key
UK_PARTITION	VARCHAR2(40)	Not used
COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that satisfy this unique key
COMPLIANT_CNT_QUERY	VARCHAR2(4000)	Not used
NONCOMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy this unique key
NONCOMPLIANT_CNT_QUERY	VARCHAR2(4000)	Not used
DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row satisfies this unique key and which does not
CHILDLESS_QUERY	VARCHAR2(4000)	Not used
UPDATED_ON	DATE	Update timestamp

Table 2–114 (Cont.) ALL_IV_PROFILE_UNIQUE_KEYS

Column Name	Data Type	Description
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.12 Data Rules Views

Table 2–115 ALL_IV_DATA_RULES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
DATARULE_ID	NUMBER(9)	ID of the data rule
DATARULE_NAME	VARCHAR2(255)	Physical name of the data rule
BUSINESS_NAME	VARCHAR2(1000)	Business name of the data rule
DATARULE_TYPE	VARCHAR2(4000)	Type of data rule
DESCRIPTION	VARCHAR2(4000)	Description of the data rule
IS_VALID	VARCHAR2(13)	Not used
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–116 ALL_IV_DATA_RULE_ATTRIBUTES

Column Name	Data Type	Description
DATARULE_ID	NUMBER(9)	ID of the data rule
DATARULE_NAME	VARCHAR2(255)	Name of the data rule
DATARULEGROUP_ID	NUMBER(9)	ID of the owning data rule group
DATARULEGROUP_NAME	VARCHAR2(255)	Name of the owning data rule group
DATARULEATTR_ID	NUMBER(9)	ID of the data rule attribute
NAME	VARCHAR2(255)	Name of the data rule attribute
BUSINESS_NAME	VARCHAR2(1000)	Business name of data rule attribute
DESCRIPTION	VARCHAR2(4000)	Description of data rule attribute
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–117 ALL_IV_DATA_RULE_ATTR_USAGES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
RELATION_ID	NUMBER(9)	ID of owning relation
RELATION_NAME	VARCHAR2(255)	Name of owning relation
DATARULE_USAGE_ID	NUMBER(9)	ID of owning data rule usage
DATARULE_USAGE_NAME	VARCHAR2(255)	Name of owning data rule usage
DATARULE_GROUP_USAGE_ID	NUMBER(9)	ID of owning data rule usage group
DATARULE_GROUP_USAGE_NAME	VARCHAR2(255)	Name of owning data rule usage group
DATARULE_ATTR_USAGE_ID	NUMBER(9)	ID of data rule attribute usage
NAME	VARCHAR2(255)	Name of data rule attribute usage
BUSINESS_NAME	VARCHAR2(1000)	Business name of data rule attribute usage
DESCRIPTION	VARCHAR2(4000)	Description of data rule attribute usage
DATARULE_SCHEMA_ID	NUMBER(9)	ID of data rule folder
DATARULE_SCHEMA_NAME	VARCHAR2(255)	Name of data rule folder
DATARULE_ID	NUMBER(9)	ID of data rule
DATARULE_NAME	VARCHAR2(255)	Name of data rule
DATARULE_GROUP_ID	NUMBER(9)	ID of data rule group
DATARULE_GROUP_NAME	VARCHAR2(255)	Name of data rule group
DATARULE_ATTR_ID	NUMBER(9)	ID of data rule attribute
DATARULE_ATTR_NAME	VARCHAR2(255)	Name of data rule attribute
REFERENCED_SCHEMA	NUMBER(9)	ID of referenced schema of date rule usage group
REFERENCE_SCHEMA_NAME	VARCHAR2(255)	Name of referenced schema of data rule usage group
REFERENCED_RELATION_ID	NUMBER(9)	ID if referenced relation of data rule usage group
REFERENCE_RELATION_NAME	VARCHAR2(255)	Name of referenced relation of data rule usage group
ATTRIBUTE_ID	NUMBER(9)	ID of the attribute
ATTRIBUTE_NAME	VARCHAR2(255)	Name of the attribute
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–118 ALL_IV_DATA_RULE_DOMAINS

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
DATARULE_ID	NUMBER(9)	ID of the data rule
DATARULE_NAME	VARCHAR2(255)	Name of the data rule
PROPERTY_ID	NUMBER(9)	ID of domain property
NAME	VARCHAR2(255)	Value of domain property
BUSINESS_NAME	VARCHAR2(1000)	Not used
DESCRIPTION	VARCHAR2(4000)	Description of the data rule
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–119 ALL_IV_DATA_RULE_GROUPS

Column Name	Data Type	Description
DATARULE_ID	NUMBER(9)	ID of the data rule
DATARULE_NAME	VARCHAR2(255)	Physical name of the data rule
DATARULEGROUP_ID	NUMBER(9)	ID of data rule group
DATARULEGROUP_NAME	VARCHAR2(255)	Name of data rule group
BUSINESS_NAME	VARCHAR2(1000)	Business name of data rule group
DESCRIPTION	VARCHAR2(4000)	Description of the data rule group
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–120 ALL_IV_DATA_RULE_GROUP_USAGES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
RELATION_ID	NUMBER(9)	ID of owning relation
RELATION_NAME	VARCHAR2(255)	Name of owning relation
DATARULE_USAGE_ID	NUMBER(9)	ID of data rule usage
DATARULE_USAGE_NAME	VARCHAR2(255)	Name of data rule usage
DATARULE_GROUP_USAGE_ID	NUMBER(9)	ID of data rule usage group
NAME	VARCHAR2(255)	Name of data rule usage group
BUSINESS_NAME	VARCHAR2(1000)	Business name of data rule usage group

Table 2–120 (Cont.) ALL_IV_DATA_RULE_GROUP_USAGES

Column Name	Data Type	Description
DESCRIPTION	VARCHAR2(4000)	Description of data rule usage group
DATARULE_SCHEMA_ID	NUMBER(9)	ID of referenced data rule folder
DATARULE_SCHEMA_NAME	VARCHAR2(255)	Name of referenced data rule folder
DATARULE_ID	NUMBER(9)	ID of referenced data rule
DATARULE_NAME	VARCHAR2(255)	Name of referenced data rule
DATARULE_GROUP_ID	NUMBER(9)	ID of referenced data rule group
DATARULE_GROUP_NAME	VARCHAR2(255)	Name of referenced data rule group
REFERENCED_SCHEMA	NUMBER(9)	ID of referenced schema
REFERENCE_SCHEMA_NAME	VARCHAR2(255)	Name of referenced schema
REFERENCED_RELATION_ID	NUMBER(9)	ID of referenced relation
REFERENCE_RELATION_NAME	VARCHAR2(255)	Name of referenced relation
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–121 ALL_IV_DATA_RULE_PROPERTIES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
DATARULE_ID	NUMBER(9)	ID of the data rule
DATARULE_NAME	VARCHAR2(255)	Physical name of the data rule
PROPERTY_ID	NUMBER(9)	ID of data rule property
NAME	VARCHAR2(255)	Name of data rule property
BUSINESS_NAME	VARCHAR2(1000)	Business name of the data rule property
VALUE	VARCHAR2(4000)	Value of data rule property
DESCRIPTION	VARCHAR2(4000)	Description of the data rule property
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–122 ALL_IV_DATA_RULE_USAGES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the owning schema
SCHEMA_NAME	VARCHAR2(255)	Name of the owning schema
RELATION_ID	NUMBER(9)	Owning relation of data rule usage

Table 2–122 (Cont.) ALL_IV_DATA_RULE_USAGES

Column Name	Data Type	Description
RELATION_NAME	VARCHAR2(255)	Name of owning relation of data rule usage
DATARULE_USAGE_ID	NUMBER(9)	ID of data rule usage
NAME	VARCHAR2(255)	Name of data rule usage
BUSINESS_NAME	VARCHAR2(1000)	Business name of data rule usage
DESCRIPTION	VARCHAR2(4000)	Description of data rule usage
DATARULE_SCHEMA_ID	NUMBER(9)	Referenced data rule folder ID
DATARULE_SCHEMA_NAME	VARCHAR2(255)	Referenced data rule folder name
DATARULE_ID	NUMBER(9)	Referenced data rule ID
DATARULE_NAME	VARCHAR2(255)	Referenced data rule name
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.13 User Defined Object Views

Table 2–123 ALL_IV_UDO_FCOS

Column Name	Data Type	Description
OWNER_ID	NUMBER(9)	ID of the owner
OWNER_NAME	VARCHAR2(255)	Name of the owner
FIRST_CLASS_OBJECT_ID	NUMBER(9)	ID of the First Class Object
FIRST_CLASS_OBJECT_NAME	VARCHAR2(255)	Name of the First Class Object
FIRST_CLASS_OBJECT_TYPE	VARCHAR2(255)	Type of the First Class Object
BUSINESS_NAME	VARCHAR2(1000)	Business name of the First Class Object
DESCRIPTION	VARCHAR2(4000)	Description of the First Class Object
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–124 ALL_IV_UDO_FOLDERS

Column Name	Data Type	Description
OWNER_ID	NUMBER(9)	ID of the owner
OWNER_NAME	VARCHAR2(255)	Name of the owner
FOLDER_ID	NUMBER(9)	ID of the folder
FOLDER_NAME	VARCHAR2(255)	Name of the folder
FOLDER_TYPE	VARCHAR2(255)	Type of folder
BUSINESS_NAME	VARCHAR2(1000)	Business name of the folder

Table 2–124 (Cont.) ALL_IV_UDO_FOLDERS

Column Name	Data Type	Description
DESCRIPTION	VARCHAR2(4000)	Description of the folder
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–125 ALL_IV_UDO_SCOS

Column Name	Data Type	Description
OWNER_ID	NUMBER(9)	ID of the owner
OWNER_NAME	VARCHAR2(255)	Name of the owner
SECOND_CLASS_OBJECT_ID	NUMBER(9)	ID of the Second Class Object
SECOND_CLASS_OBJECT_NAME	VARCHAR2(255)	Name of the Second Class Object
SECOND_CLASS_OBJECT_TYPE	VARCHAR2(255)	Type of Second Class Object
BUSINESS_NAME	VARCHAR2(1000)	Business name of the Second Class Object
DESCRIPTION	VARCHAR2(4000)	Description of the Second Class Object
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–126 ALL_IV_UDO_ASSOCIATIONS

Column Name	Data Type	Description
OWNER_ID	NUMBER(9)	ID of the owner
OWNER_NAME	VARCHAR2(255)	Name of the owner
OWNER_TYPE	VARCHAR2(255)	Type of the owner
ASSOCIATION_ID	NUMBER(9)	ID of the association
SOURCE_ROLE	VARCHAR2(255)	Role of the owner
TARGET_ROLE	VARCHAR2(255)	Role of the associated object
TARGET_ID	NUMBER(9)	ID of the target
TARGET_NAME	VARCHAR2(255)	Name of the target
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.14 Expert Views

Table 2–127 ALL_IV_EXPERTS

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
EXPERT_ID	NUMBER(9)	ID of the expert
EXPERT_NAME	VARCHAR2(255)	Name of the expert
BUSINESS_NAME	VARCHAR2(1000)	Business name of the expert
DESCRIPTION	VARCHAR2(4000)	Description of the expert
BOUND_OBJECT_ID	NUMBER(9)	ID of the bound object
BOUND_OBJECT_NAME	VARCHAR2(40)	Name of the bound object
IS_VALID	VARCHAR2(13)	Is this expert valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–128 ALL_IV_EXPERT_PARAMETERS

Column Name	Data Type	Description
PARAMETER_OWNER_ID	NUMBER(9)	ID of the parameter owner
PARAMETER_OWNER_NAME	VARCHAR2(255)	Name of the parameter owner
PARAMETER_OWNER_TYPE	VARCHAR2(6)	Type of the parameter owner
PARAMETER_ID	NUMBER(9)	ID of the parameter
PARAMETER_NAME	VARCHAR2(255)	Name of the parameter
BUSINESS_NAME	VARCHAR2(1000)	Business name of the parameter
DESCRIPTION	VARCHAR2(4000)	Description of the parameter
POSITION	NUMBER(9)	The position of the parameter
DATA_TYPE	VARCHAR2(40)	Data type for this parameter
DEFAULT_VALUE	VARCHAR2(4000)	Default value
DIRECTION	VARCHAR2(5)	Direction of this parameter
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–129 ALL_IV_EXPERT_TASKS

Column Name	Data Type	Description
EXPERT_ID	NUMBER(9)	ID of the expert
EXPERT_NAME	VARCHAR2(255)	Name of the expert
TASK_ID	NUMBER(9)	ID of the task
TASK_NAME	VARCHAR2(255)	Name of the task
BUSINESS_NAME	VARCHAR2(1000)	Business name of the task
DESCRIPTION	VARCHAR2(4000)	Description of the task
TASK_TYPE	VARCHAR2(4000)	Type of the task
BOUND_OBJECT_ID	NUMBER(9)	ID of the bound object
BOUND_OBJECT_NAME	VARCHAR2(40)	Name of the bound object
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user
MAIN	VARCHAR2(4000)	The main processing script of the task
PREPROCESSING	VARCHAR2(4000)	The pre-processing script of the task
POSTPROCESSING	VARCHAR2(4000)	The post-processing script of the task
INSTRUCTION	VARCHAR2(4000)	Instruction for running the task

Table 2–130 ALL_IV_EXPERT_TRANSITIONS

Column Name	Data Type	Description
EXPERT_ID	NUMBER(9)	ID of the expert
EXPERT_NAME	VARCHAR2(255)	Name of the expert
TRANSITION_ID	NUMBER(9)	ID of the transition
TRANSITION_NAME	VARCHAR2(255)	Name of the transition
BUSINESS_NAME	VARCHAR2(1000)	Business name of the transition
DESCRIPTION	VARCHAR2(4000)	Description of the transition
CONDITION	VARCHAR2(4000)	Transition condition
TRANSITION_ORDER	NUMBER(9)	Transition order
SOURCE_ACTIVITY_ID	NUMBER(9)	ID of the source task
SOURCE_ACTIVITY_NAME	VARCHAR2(255)	Name of the source task
TARGET_ACTIVITY_ID	NUMBER(9)	ID of the target task
TARGET_ACTIVITY_NAME	VARCHAR2(255)	Name of the target task
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–131 ALL_IV_EXPERT_VARIABLES

Column Name	Data Type	Description
EXPERT_ID	NUMBER(9)	ID of the expert
EXPERT_NAME	VARCHAR2(255)	Name of the expert
VARIABLE_ID	NUMBER(9)	ID of the variable
VARIABLE_NAME	VARCHAR2(255)	Name of the variable
BUSINESS_NAME	VARCHAR2(1000)	Business name of the variable
DESCRIPTION	VARCHAR2(4000)	Description of the variable
POSITION	NUMBER(9)	Position of the variable
DATA_TYPE	VARCHAR2(40)	Data type of the variable
DEFAULT_VALUE	VARCHAR2(4000)	Default value of the variable
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–132 ALL_IV_NESTED_EXPERTS

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
PARENT_EXPERT_ID	NUMBER(9)	ID of the parent expert
PARENT_EXPERT_NAME	VARCHAR2(255)	Name of the parent expert
EXPERT_ID	NUMBER(9)	ID of the expert
EXPERT_NAME	VARCHAR2(255)	Name of the expert
BUSINESS_NAME	VARCHAR2(1000)	Business name of the expert
DESCRIPTION	VARCHAR2(4000)	Description of the expert
BOUND_OBJECT_ID	NUMBER(9)	ID of the bound object
BOUND_OBJECT_NAME	VARCHAR2(40)	Name of the bound object
IS_VALID	VARCHAR2(13)	Is this nested expert valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.15 Business Intelligence Views

Table 2–133 ALL_IV_ALTERNATIVE_SORT_ORDERS

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
ALT_SORT_ORDER_ID	NUMBER(9)	ID of the alternative sort order
ALT_SORT_ORDER_NAME	VARCHAR2(255)	Name of the alternative sort order
BUSINESS_NAME	VARCHAR2(1000)	Business name of the alternative sort order
DESCRIPTION	VARCHAR2(4000)	Description of the alternative sort order
IS_DRILL_TO_DETAIL	CHAR(1)	Whether it acts as a Drill to Detail when deployed to Discoverer
IS_CACHE_VALUES	CHAR(1)	Indicates whether Discoverer should cache the list of values in memory for the current session, once it has been displayed for the first time
IS_REQUIRE_SEARCH	CHAR(1)	Causes Discoverer to request the end user to enter search criteria to reduce the list of values to a subset of the whole list
IS_SHOW_IN_NAVIGATOR	CHAR(1)	Indicates that Discoverer should show the values in the "Select Items" page of the Worksheet Wizard (the item navigator).
IS_SORTED_DISTINCT	CHAR(1)	Indicates that the values should be displayed alphabetically sorted and with duplicates hidden
RETRIEVE_VALUE_GROUP_SIZE	NUMBER(9)	The maximum number of rows to be fetched from the database at a time
VALUES_ITEM_ID	NUMBER(9)	The ID of the item that contains the values to be sorted
VALUES_ITEM_NAME	VARCHAR2(255)	The name of the item that contains the values to be sorted
ORDER_ITEM_ID	NUMBER(9)	The identifier of the item that defines the order in which the values in the VALUES_ITEM_ID field are to be sorted
ORDER_ITEM_NAME	VARCHAR2(255)	The name of the item that defines the order in which the values in the VALUES_ITEM_ID field are to be sorted
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–134 ALL_IV_BUSINESS_AREAS

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
BUSINESS_AREA_ID	NUMBER(9)	Id of the business area

Table 2–134 (Cont.) ALL_IV_BUSINESS_AREAS

Column Name	Data Type	Description
BUSINESS_AREA_NAME	VARCHAR2(255)	Name of the business area
BUSINESS_NAME	VARCHAR2(1000)	Business name of the business area
DESCRIPTION	VARCHAR2(4000)	Description of the business area
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–135 ALL_IV_BUSINESS_AREA_FOLDERS

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
BUSINESS_AREA_ID	NUMBER(9)	ID of the business area
BUSINESS_AREA_NAME	VARCHAR2(255)	Name of the business area
ITEM_FOLDER_ID	NUMBER(9)	Identifier of the Item Folder present in the Business Area
ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the item folder present in the business area
BUSINESS_NAME	VARCHAR2(1000)	Business name of the item folder present in the business area
DESCRIPTION	VARCHAR2(4000)	Description of the item folder present in the business area
FOLDER_TYPE	VARCHAR2(40)	The item folder type (simple or complex) of the item folder present in the business area
IS_VISIBLE	NUMBER	The visibility of the item folder to the end-user
VALID	CHAR(1)	Flag to indicate if it is valid
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–136 ALL_IV_PRESENTATION_TEMPLATES

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
TEMPLATE_ID	NUMBER(9)	ID of the template
TEMPLATE_NAME	VARCHAR2(255)	Name of the template
BUSINESS_NAME	VARCHAR2(1000)	Business name of the presentation template
DESCRIPTION	VARCHAR2(4000)	Description of the presentation template

Table 2–136 (Cont.) ALL_IV_PRESENTATION_TEMPLATES

Column Name	Data Type	Description
PRESENTATION_TYPE	VARCHAR2(40)	The presentation type (CROSSTAB, PIE, BAR_VERT_CLUST)
CUBE_ID	NUMBER(9)	ID of the cube that is referenced in the presentation template
CUBE_NAME	VARCHAR2(255)	Name of the cube that is referenced in the presentation template
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–137 ALL_IV_DRILLS_TO_DETAIL

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
DRILL_TO_DETAIL_ID	NUMBER(9)	ID of the drill to detail
DRILL_TO_DETAIL_NAME	VARCHAR2(255)	Name of the drill to detail
BUSINESS_NAME	VARCHAR2(1000)	Business name of the drill to detail
DESCRIPTION	VARCHAR2(4000)	Description of the drill to detail
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–138 ALL_IV_DRILL_LEVELS

Column Name	Data Type	Description
DRILL_PATH_ID	NUMBER(9)	The identifier of the owning drill path
DRILL_PATH_NAME	VARCHAR2(255)	The name of the owning drill path
DRILL_LEVEL_ID	NUMBER(9)	The identifier of the drill level
DRILL_LEVEL_NAME	VARCHAR2(255)	The name of the drill level
BUSINESS_NAME	VARCHAR2(1000)	The business name of the drill level
DESCRIPTION	VARCHAR2(4000)	The description of the drill level
IS_DEFAULT_ROOT_LEVEL	VARCHAR2(1)	Is it the default root level
IS_GRAND_TOTAL_LEVEL	VARCHAR2(1)	Is it the grand total level
SUPPORTS_ROLLUP	VARCHAR2(1)	Flag to indicate if it supports rollup
PARENT_DRILL_LEVEL_ID	NUMBER(9)	The identifier of the parent level in the drill path hierarchy
PARENT_DRILL_LEVEL_NAME	VARCHAR2(255)	The name of the parent level in the drill path hierarchy

Table 2–138 (Cont.) ALL_IV_DRILL_LEVELS

Column Name	Data Type	Description
RELATED_LEVEL_ID	NUMBER(9)	The identifier of the hierarchy level that the drill level was derived from
RELATED_LEVEL_NAME	VARCHAR2(255)	The name of the hierarchy level that the drill level was derived from
ITEM_OWNER_ID	NUMBER(9)	ID of the item owner
ITEM_OWNER_NAME	VARCHAR2(1000)	Name of the item owner
ITEM_FOLDER_ID	NUMBER(9)	The identifier of the item folder that the drill level is based on
ITEM_FOLDER_NAME	VARCHAR2(255)	The name of the item folder that the drill level is based on
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–139 ALL_IV_ITEM_FOLDERS

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
ITEM_FOLDER_ID	NUMBER(9)	ID of the item folder
ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the item folder
BUSINESS_NAME	VARCHAR2(1000)	Business name of the item folder
DESCRIPTION	VARCHAR2(4000)	Description of the item folder
FOLDER_TYPE	VARCHAR2(40)	Type of item folder (simple or complex)
IS_VISIBLE	NUMBER	Whether the item folder is visible to the end user
SOURCE_OBJECT_ID	NUMBER(9)	ID of the source object
SOURCE_OBJECT_TYPE	VARCHAR2(4000)	Type of the source object (Table, Cube, Dimension)
SOURCE_OBJECT_NAME	VARCHAR2(255)	Name of the source object
DIMENSION_ROLE_ID	NUMBER(9)	If this item folder was derived for a dimension role, then this returns the identifier of that dimension role
DIMENSION_ROLE_NAME	VARCHAR2(255)	If this item folder was derived for a dimension role, then this returns the name of that dimension role
RELATED_LEVEL_ID	NUMBER(9)	If this item folder was derived for a level, then this returns the identifier of the level
RELATED_LEVEL_NAME	VARCHAR2(255)	If this item folder was derived for a level then this returns the name of the level
VALID	CHAR(1)	Flag to indicate if it is valid
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp

Table 2–139 (Cont.) ALL_IV_ITEM_FOLDERS

Column Name	Data Type	Description
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–140 ALL_IV_ITEM_FOLDER_JOIN_USAGES

Column Name	Data Type	Description
ITEM_FOLDER_ID	NUMBER(9)	ID of the complex item folder
ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the complex item folder
JOIN_ID	NUMBER(9)	ID of the join between two base item folders of the complex item folder
JOIN_NAME	VARCHAR2(255)	Name of the join between two base item folders of the complex item folder

Table 2–141 ALL_IV_ITEMS

Column Name	Data Type	Description
ITEM_OWNER_ID	NUMBER(9)	Item owner ID
ITEM_OWNER_NAME	VARCHAR2(1000)	Name of the item owner
ITEM_FOLDER_ID	NUMBER(9)	ID of the item folder
ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the item folder
ITEM_ID	NUMBER(9)	ID of the item
ITEM_NAME	VARCHAR2(255)	Name of the item
BUSINESS_NAME	VARCHAR2(1000)	Business name of the item
DESCRIPTION	VARCHAR2(4000)	Description of the item
IS_COLUMN_ITEM	CHAR(1)	Whether this item is based on a column
DATATYPE	VARCHAR2(40)	The data type of the item
ALIGNMENT	VARCHAR2(40)	Alignment for display (Default, Left, Center, Right)
CASE_STORAGE	VARCHAR2(40)	How alphabetic characters are stored in the database (Unknown, Lower, Upper, Mixed)
CONTENT_TYPE	VARCHAR2(40)	Specifies whether the item can be used to launch an external application
DEFAULT_AGGREGATE	VARCHAR2(255)	Name of default rollup function
DEFAULT_POSITION	VARCHAR2(40)	Default placement for query item (Unknown, Measure, Axis, X-axis, Y-axis, Z-axis)
DEFAULT_WIDTH	NUMBER(9)	Default number of characters in display
DISPLAY_CASE	VARCHAR2(40)	How alphabetic characters should be displayed (Unchanged, Lower, Upper, InitCapped)
FORMAT_MASK	VARCHAR2(255)	The format of the way that the item is displayed

Table 2–141 (Cont.) ALL_IV_ITEMS

Column Name	Data Type	Description
FORMULA	CLOB	The item's expression
HEADING	VARCHAR2(255)	The value of the default heading in a report
IS_VISIBLE	NUMBER	Whether the item is visible to the end user
IS_WORD_WRAP	CHAR(1)	Whether word wrap is allowed when displaying values in a report
MAX_CHAR_FETCHED	NUMBER(9)	Maximum number of characters retrieved from the database
REPLACE_NULL_WITH	VARCHAR2(255)	Value to be displayed for null values
RELATED_ATTRIBUTE_ID	NUMBER(9)	If this item was derived, then the identifier of the attribute it was derived from
RELATED_ATTRIBUTE_TYPE	VARCHAR2(4000)	If this item was derived, then the type of the attribute it was derived from
RELATED_ATTRIBUTE_NAME	VARCHAR2(255)	If this item was derived, then the name of the attribute it was derived from
LIST_OF_VALUES_ID	NUMBER	If this item has a list of values, then the list of values identifier
LIST_OF_VALUES_NAME	VARCHAR2(255)	If this item has a list of values, then the list of values name
ALTERNATIVE_SORT_ORDER_ID	NUMBER	If this item has an alternative sort order then the ID of the alternative sort order
ALTERNATIVE_SORT_ORDER_NAME	VARCHAR2(255)	If this item has an alternative sort order then the name of the alternative sort order
DRILL_TO_DETAIL_ID	NUMBER	If this item has a drill to detail then the ID of the drill to detail
DRILL_TO_DETAIL_NAME	VARCHAR2(255)	If this item has a drill to detail then the name of the drill to detail
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–142 ALL_IV_ITEM_FORMULA_REFS

Column Name	Data Type	Description
ITEM_OWNER_ID	NUMBER(9)	ID of item owner
ITEM_OWNER_NAME	VARCHAR2(1000)	Name of the item owner
ITEM_FOLDER_ID	NUMBER(9)	ID of the item folder
ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the item folder
ITEM_ID	NUMBER(9)	ID of the item
ITEM_NAME	VARCHAR2(255)	Name of the item
TAG	NUMBER(9)	Refers to the tag value used in the formula of the item
REFERENCED_FUNCTION_ID	NUMBER(9)	ID of the referenced function

Table 2–142 (Cont.) ALL_IV_ITEM_FORMULA_REFS

Column Name	Data Type	Description
REFERENCED_FUNCTION_TYPE	VARCHAR2(4000)	Type of the referenced function
REFERENCED_FUNCTION_NAME	VARCHAR2(255)	Name of the referenced function
REFERENCED_ITEM_OWNER_ID	NUMBER(9)	ID of the referenced item owner
REFERENCED_ITEM_OWNER_NAME	VARCHAR2(1000)	Name of the referenced item owner
REFERENCED_ITEM_FOLDER_ID	NUMBER(9)	ID of the referenced item folder
REFERENCED_ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the referenced item folder
REFERENCED_ITEM_ID	NUMBER(9)	ID of the referenced item
REFERENCED_ITEM_NAME	VARCHAR2(255)	Name of the referenced item

Table 2–143 ALL_IV_DATA_ITEMS

Column Name	Data Type	Description
TEMPLATE_ID	NUMBER(9)	ID of the presentation template
TEMPLATE_NAME	VARCHAR2(255)	Name of the presentation template
DATA_ITEM_ID	NUMBER(9)	ID of the data item
DATA_ITEM_NAME	VARCHAR2(255)	Name of the data item
BUSINESS_NAME	VARCHAR2(1000)	Business name of the data item
DESCRIPTION	VARCHAR2(4000)	Description of the data item
MEASURE_ID	NUMBER(9)	The identifier of the measure used as the data item
MEASURE_NAME	VARCHAR2(255)	The name of the measure used as the data item
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–144 ALL_IV_EDGE_ITEMS

Column Name	Data Type	Description
TEMPLATE_ID	NUMBER(9)	ID of the presentation template
TEMPLATE_NAME	VARCHAR2(255)	Name of the presentation template
EDGE_ITEM_ID	NUMBER(9)	Identifier of the edge item
EDGE_ITEM_NAME	VARCHAR2(255)	Name of the edge item
BUSINESS_NAME	VARCHAR2(1000)	Business name of the edge item
DESCRIPTION	VARCHAR2(4000)	Description of the edge item
PLACEMENT	VARCHAR2(40)	The axis the edge item is on
DIMENSION_ROLE_ID	NUMBER(9)	ID of the dimension role
DIMENSION_ROLE_NAME	VARCHAR2(255)	Name of the dimension role
DIMENSION_ID	NUMBER(9)	ID of the dimension
DIMENSION_NAME	VARCHAR2(255)	Name of the dimension

Table 2–144 (Cont.) ALL_IV_EDGE_ITEMS

Column Name	Data Type	Description
HIERARCHY_ID	NUMBER(9)	Not used
HIERARCHY_NAME	VARCHAR2(255)	Not used
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–145 ALL_IV_DRILL_PATHS

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
DRILL_PATH_ID	NUMBER(9)	ID of the drill path
DRILL_PATH_NAME	VARCHAR2(255)	Name of the drill path
BUSINESS_NAME	VARCHAR2(1000)	Business name of the drill path
DESCRIPTION	VARCHAR2(4000)	Description of the drill path
DIMENSION_ID	NUMBER(9)	If the drill path was derived from a dimension hierarchy, then the ID of the dimension
DIMENSION_NAME	VARCHAR2(255)	If the drill path was derived from a dimension hierarchy then the name of the dimension
DIMENSION_ROLE_ID	NUMBER(9)	If the drill path was derived from a dimension role's hierarchy then its identifier
DIMENSION_ROLE_NAME	VARCHAR2(255)	If the drill path was derived from a dimension role's hierarchy then its name
HIERARCHY_ID	NUMBER(9)	The identifier of the hierarchy the drill path was derived from
HIERARCHY_NAME	VARCHAR2(255)	The name of the hierarchy the drill path was derived from
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–146 ALL_IV_DRILL_LEVEL_ITEMS

Column Name	Data Type	Description
DRILL_PATH_ID	NUMBER(9)	Owning drill path identifier
DRILL_PATH_NAME	VARCHAR2(255)	Owning drill path name
DRILL_LEVEL_ID	NUMBER(9)	ID of the drill level
DRILL_LEVEL_NAME	VARCHAR2(255)	Name of the drill level
DRILL_ITEM_ID	NUMBER(9)	ID of the drill item

Table 2–146 (Cont.) ALL_IV_DRILL_LEVEL_ITEMS

Column Name	Data Type	Description
DRILL_ITEM_NAME	VARCHAR2(1000)	Name of the drill item
BUSINESS_NAME	VARCHAR2(1000)	Business name of drill level item
DESCRIPTION	VARCHAR2(4000)	Description
ITEM_OWNER_ID	NUMBER(9)	ID of the item owner
ITEM_OWNER_NAME	VARCHAR2(1000)	Name of the item owner
ITEM_FOLDER_ID	NUMBER(9)	Item folder ID
ITEM_FOLDER_NAME	VARCHAR2(255)	Item folder name
ITEM_ID	NUMBER(9)	ID of the item
ITEM_NAME	VARCHAR2(255)	Name of the item
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–147 ALL_IV_DRILL_PATH_JOIN_USAGES

Column Name	Data Type	Description
DRILL_PATH_ID	NUMBER(9)	ID of the drill path
DRILL_PATH_NAME	VARCHAR2(255)	Name of the drill path
PARENT_DRILL_LEVEL_ID	NUMBER(9)	ID of the parent drill level
PARENT_DRILL_LEVEL_ NAME	VARCHAR2(255)	Name of the parent drill level
CHILD_DRILL_LEVEL_ID	NUMBER(9)	ID of the child drill level
CHILD_DRILL_LEVEL_NAME	VARCHAR2(255)	Name of the child drill level
JOIN_ID	NUMBER(9)	ID of the join
JOIN_NAME	VARCHAR2(255)	Name of the join

Table 2–148 ALL_IV_LISTS_OF_VALUES

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
LIST_OF_VALUES_ID	NUMBER(9)	ID of the list of values
LIST_OF_VALUES_NAME	VARCHAR2(255)	Name of the list of values
BUSINESS_NAME	VARCHAR2(1000)	Business name of the list of values
DESCRIPTION	VARCHAR2(4000)	Description of the list of values
IS_DRILL_TO_DETAIL	CHAR(1)	Whether it acts as a drill to detail when deployed to Discoverer
IS_CACHE_VALUES	CHAR(1)	Indicates whether Discoverer should cache the list of values in memory for the current session, once it has been displayed for the first time

Table 2–148 (Cont.) ALL_IV_LISTS_OF_VALUES

Column Name	Data Type	Description
IS_REQUIRE_SEARCH	CHAR(1)	Value 1 causes Discoverer to request the end user to enter search criteria to reduce the list of values to a subset of the whole list
IS_SHOW_IN_NAVIGATOR	CHAR(1)	Indicates whether Discoverer should show the values in the "Select Items" page of the Worksheet Wizard (the item navigator)
IS_SORTED_DISTINCT	CHAR(1)	Value 1 indicates that the values should be displayed alphabetically sorted and with duplicates hidden
RETRIEVE_VALUE_GROUP_SIZE	NUMBER(9)	The maximum number of rows to be fetched from the database at a time
VALUES_ITEM_ID	NUMBER(9)	The identifier of the item that supplies the values
VALUES_ITEM_NAME	VARCHAR2(255)	The name of the item that supplies the values
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–149 ALL_IV_REGISTERED_FUNCTIONS

Column Name	Data Type	Description
MODULE_ID	NUMBER(9)	ID of the module
MODULE_NAME	VARCHAR2(255)	Name of the module
REGISTERED_FUNCTION_ID	NUMBER(9)	ID of the registered function
REGISTERED_FUNCTION_NAME	VARCHAR2(255)	Name of the registered function
BUSINESS_NAME	VARCHAR2(1000)	Business name of the registered function
DESCRIPTION	VARCHAR2(4000)	Description of the registered function
SIGNATURE	VARCHAR2(4000)	The signature of the registered function
IS_AVAILABLE	CHAR(1)	Whether a Discoverer end user may use this function in calculations
SOURCE_FUNCTION_ID	NUMBER(9)	The identifier of the function it was derived from
SOURCE_FUNCTION_NAME	VARCHAR2(255)	The name of the function it was derived from
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–150 ALL_IV_CONDITION_FORMULA_REFS

Column Name	Data Type	Description
ITEM_FOLDER_ID	NUMBER(9)	ID of the item folder
ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the item folder
ITEM_ID	NUMBER(9)	ID of the condition
ITEM_NAME	VARCHAR2(255)	Name of the condition
TAG	NUMBER(9)	Refers to the tag value used in the condition's formula
REFERENCED_FUNCTION_ID	NUMBER(9)	ID of the referenced function
REFERENCED_FUNCTION_TYPE	VARCHAR2(4000)	Type of the referenced function
REFERENCED_ITEM_ID	NUMBER(9)	ID of the referenced item
REFERENCED_ITEM_NAME	VARCHAR2(255)	Name of the referenced item
REFERENCED_CONDITION_ID	NUMBER(9)	ID of the referenced condition
REFERENCED_CONDITION_NAME	VARCHAR2(255)	Name of the referenced condition

Table 2–151 ALL_IV_JOIN_COMPONENTS

Column Name	Data Type	Description
ITEM_OWNER_ID	NUMBER(9)	ID of the item owner
ITEM_OWNER_NAME	VARCHAR2(1000)	Name of the item owner
ITEM_FOLDER_ID	NUMBER(9)	ID of the item folder
ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the item folder
JOIN_ID	NUMBER(9)	ID of the join
JOIN_NAME	VARCHAR2(255)	Name of the join
JOIN_COMPONENT_ID	NUMBER(9)	Identifier of the join component
JOIN_COMPONENT_NAME	VARCHAR2(255)	Name of the join component
BUSINESS_NAME	VARCHAR2(1000)	Business Name of the join component
DESCRIPTION	VARCHAR2(4000)	Description of the join component
JOIN_OPERATOR	VARCHAR2(40)	The operator for the join component
DETAIL_ITEM_ID	NUMBER(9)	The identifier of the item referenced in the detail item folder
DETAIL_ITEM_NAME	VARCHAR2(255)	The name of the item referenced in the detail item folder
MASTER_ITEM_ID	NUMBER(9)	The identifier of the item referenced in the master item folder
MASTER_ITEM_NAME	VARCHAR2(255)	The name of the item referenced in the master item folder
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–152 ALL_IV_JOINS

Column Name	Data Type	Description
ITEM_FOLDER_ID	NUMBER(9)	ID of the item owner
ITEM_OWNER_NAME	VARCHAR2(1000)	Name of the item owner
ITEM_FOLDER_ID	NUMBER(9)	ID of the detail item folder
ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the detail item folder
JOIN_ID	NUMBER(9)	ID of the join
JOIN_NAME	VARCHAR2(255)	Name of the join
BUSINESS_NAME	VARCHAR2(1000)	Business name of the join
DESCRIPTION	VARCHAR2(4000)	Description of the join
IS_OUTER_JOIN_ON_MASTER	CHAR(1)	Indicates whether to perform an outer join on the master item folder. If this is set, any detail rows that have no related master row will be included in the results of the join
IS_OUTER_JOIN_ON_DETAIL	CHAR(1)	Indicates whether to perform an outer join on the detail item folder. If this is set, any master rows that have no related detail rows will be included in the results of the join
IS_DETAIL_EXISTS_ON_MASTER	CHAR(1)	Indicates whether every detail row must refer to a valid master row
IS_ONE_TO_ONE	CHAR(1)	Indicates whether each master row only ever has a single detail row
REFERENCED_ITEM_OWNER_ID	NUMBER(9)	ID of the referenced item owner
REFERENCED_ITEM_OWNER_NAME	VARCHAR2(1000)	Name of the referenced item owner
REFERENCED_ITEM_FOLDER_ID	NUMBER(9)	The identifier of the master item folder referenced from the join
REFERENCED_ITEM_FOLDER_NAME	VARCHAR2(255)	The name of the master item folder referenced from the join
RELATED_FOREIGN_KEY_ID	NUMBER(9)	The identifier of the foreign key that this join was derived from
RELATED_FOREIGN_KEY_NAME	VARCHAR2(255)	The name of the foreign key that this join was derived from
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–153 ALL_IV_CONDITIONS

Column Name	Data Type	Description
ITEM_FOLDER_ID	NUMBER(9)	ID of the item folder
ITEM_FOLDER_NAME	VARCHAR2(255)	Name of the item folder
CONDITION_ID	NUMBER(9)	ID of the condition
CONDITION_NAME	VARCHAR2(255)	Name of the condition
BUSINESS_NAME	VARCHAR2(1000)	Business name of the condition

Table 2–153 (Cont.) ALL_IV_CONDITIONS

Column Name	Data Type	Description
DESCRIPTION	VARCHAR2(4000)	Description of the condition
IS_MANDATORY	CHAR(1)	Whether the condition is mandatory
FORMULA	CLOB	Formula of the condition
IS_MATCH_CASE	CHAR(1)	Whether the alphabetic character case must match exactly
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–154 ALL_IV_CATALOG_FOLDER_DRILLS

Column Name	Datatype	Description
MODULE_ID	NUMBER (9)	Module ID
MODULE_NAME	VARCHAR2 (1000)	Name of the module
CATALOG_FOLDER_ID	NUMBER (9)	ID of the catalog folder
CATALOG_FOLDER_NAME	VARCHAR2 (1000)	Name of the catalog folder
DRILL_PATH_ID	NUMBER (9)	ID of the drill path
DRILL_PATH_NAME	VARCHAR2 (1000)	Name of the drill path
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the drill path
DESCRIPTION	VARCHAR2 (4000)	Description of the drill path
IS_TIME_DIMENSION	CHAR (1)	Is time dimension
VALID	CHAR (1)	flag to indicate if it is valid
IS_VALID	VARCHAR2 (13)	Is valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–155 ALL_IV_CATALOG_FOLDER_TABLES

Column Name	Datatype	Description
MODULE_ID	NUMBER (9)	ID of the module
MODULE_NAME	VARCHAR2 (1000)	Name of the module
CATALOG_FOLDER_ID	NUMBER (9)	ID of the catalog folder
CATALOG_FOLDER_NAME	VARCHAR2 (1000)	Name of the catalog folder
LOGICAL_TABLE_ID	NUMBER (9)	ID of the logical table
LOGICAL_TABLE_NAME	VARCHAR2 (1000)	Name of the logical table
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the table
DESCRIPTION	VARCHAR2 (4000)	Description of the table

Table 2–155 (Cont.) ALL_IV_CATALOG_FOLDER_TABLES

Column Name	Datatype	Description
IS_VISIBLE	NUMBER	Is visible
IS_BRIDGE_TABLE	CHAR (1)	Is bridge table
DISTINCT_VALUES	CHAR (1)	Distinct values
VALID	CHAR (1)	Flag to indicate if valid
IS_VALID	VARCHAR (13)	Is Valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–156 ALL_IV_CATALOG_FOLDERS

Column Name	Datatype	Description
MODULE_ID	NUMBER (9)	ID of the module
MODULE_NAME	VARCHAR2 (1000)	Name of the module
CATALOG_FOLDER_ID	NUMBER (9)	ID of the catalog folder
CATALOG_FOLDER_NAME	VARCHAR2 (1000)	Name of the catalog folder
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the catalog folder
DESCRIPTION	VARCHAR2 (4000)	Description of the catalog folder
VALID	CHAR (1)	Flag to indicate if its valid
IS_VALID	VARCHAR2 (13)	Is valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–157 ALL_IV_DRILL_LEVEL_KEY_ITEMS

Column Name	Datatype	Description
DRILL_PATH_ID	NUMBER(9)	ID of the Drill Path
DRILL_PATH_NAME	VARCHAR2(1000)	Name of the Drill Path
DRILL_LEVEL_ID	NUMBER(9)	Drill Level ID
DRILL_LEVEL_NAME	VARCHAR2(1000))	Drill Level name
DRILL_LEVEL_KEY_ID	NUMBER(9)	ID of the Drill Level Key
DRILL_LEVEL_KEY_NAME	VARCHAR2(1000)	Name of the Drill Level Key
DRILL_ITEM_ID	NUMBER(9)	ID of the Drill Item
DRILL_ITEM_NAME	VARCHAR2(1000)	Name of the Drill Item

Table 2–158 ALL_IV_DRILL_LEVEL_LINKS

Column Name	Datatype	Description
DRILL_PATH_ID	NUMBER(9)	ID of the Drill Path
DRILL_PATH_NAME	VARCHAR2(1000)	Name of the Drill Path
PARENT_DRILL_LEVEL_ID	NUMBER(9)	ID of the parent Drill Level
PARENT_DRILL_LEVEL_NAME	VARCHAR2(1000)	Name of the parent Drill Level
CHILD_DRILL_LEVEL_ID	NUMBER(9)	ID of the child Drill Level
CHILD_DRILL_LEVEL_NAME	VARCHAR2(1000)	Name of the Child Drill Level

Table 2–159 ALL_IV_DRILL_LEVEL_KEYS

Column Name	Datatype	Description
DRILL_PATH_ID	NUMBER(9)	ID of the Drill Path
DRILL_PATH_NAME	VARCHAR2(1000)	Name of the Drill Path
DRILL_LEVEL_ID	NUMBER(9)	ID of the Drill Level
DRILL_LEVEL_NAME	VARCHAR2(1000)	Name of the Drill Level
DRILL_LEVEL_KEY_ID	NUMBER(9)	ID of the Drill Level Key
DRILL_LEVEL_KEY_NAME	VARCHAR2(1000)	Name of the Drill Level Key
BUSINESS_NAME	VARCHAR2(1000)	Business name of the Drill Level Key
DESCRIPTION	VARCHAR2(4000)	Description of the Drill Level Key
IS_PRIMARY_KEY	CHAR(1)	Flag to indicate if it is the primary key
IS_CHRONOLOGICAL_KEY	CHAR(1)	Is chronological key
USE_FOR_DRILLDOWN	CHAR(1)	Use for drilldown
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–160 ALL_IV_DRILL_LEVELS_DISTINCT

Column Name	Datatype	Description
DRILL_PATH_ID	NUMBER(9)	ID of the Drill Path
DRILL_PATH_NAME	VARCHAR2(1000)	Name of the Drill Path
DRILL_LEVEL_ID	NUMBER(9)	ID of the Drill Level
DRILL_LEVEL_NAME	VARCHAR2(1000)	Name of the Drill Level
BUSINESS_NAME	VARCHAR2(1000)	Business name
DESCRIPTION	VARCHAR(4000)	Description
IS_DEFAULT_ROOT_LEVEL	VARCHAR2(1)	Is default root level
IS_GRAND_TOTAL_LEVEL	VARCHAR2(1)	Is grand total level
SUPPORTS_ROLLUP	VARCHAR2(1)	Support rollup
RELATED_LEVEL_ID	NUMBER(9)	Related level ID

Table 2–160 (Cont.) ALL_IV_DRILL_LEVELS_DISTINCT

Column Name	Datatype	Description
RELATED_LEVEL_NAME	VARCHAR2(1000)	Related level name
ITEM_OWNER_ID	NUMBER(9)	ID of the Item owner
ITEM_OWNER_NAME	VARCHAR2(1000)	Name of the Item owner
ITEM_FOLDER_ID	NUMBER(9)	ID of the Item Folder
ITEM_FOLDER_NAME	VARCHAR2(1000)	Name of the Item Folder
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–161 ALL_IV_LOGICAL_TABLE_DBOJECTS

Column Name	Datatype	Description
LOGICAL_TABLE_ID	NUMBER(9)	ID of the logical table
LOGICAL_TABLE_NAME	VARCHAR2(1000)	Name of the logical table
OBJECT_ID	NUMBER(9)	ID of the object
OBJECT_TYPE	VARCHAR2(4000)	Type of object
OBJECT_NAME	VARCHAR2(1000)	Name of the object

Table 2–162 ALL_IV_LOGICAL_TABLES

Column Name	Datatype	Description
MODULE_ID	NUMBER(9)	Module ID
MODULE_NAME	VARCHAR2(1000)	Name of the module
LOGICAL_TABLE_ID	NUMBER(9)	ID of the logical table
LOGICAL_TABLE_NAME	VARCHAR2(1000)	Name of the logical table
BUSINESS_NAME	VARCHAR2(1000)	Business name of the logical table
DESCRIPTION	VARCHAR2(4000)	Description of the logical table
IS_VISIBLE	NUMBER	Is visible
IS_BRIDGE_TABLE	CHAR(1)	Is bridge table
DISTINCT_VALUES	CHAR(1)	Distinct values
SOURCE_OBJECT_ID	NUMBER(9)	ID of the source object
SOURCE_OBJECT_TYPE	VARCHAR2(4000)	Type of source object
SOURCE_OBJECT_NAME	VARCHAR2(1000)	Source object name
DIMENSION_ROLE_ID	NUMBER(9)	ID of the dimension role
DIMENSION_ROLE_NAME	VARCHAR2(1000)	Name of dimension role
RELATED_LEVEL_ID	NUMBER(9)	Related level ID
RELATED_LEVEL_NAME	VARCHAR2(1000)	Related level name
VALID	CHAR(1)	Flag to indicate if valid

Table 2–162 (Cont.) ALL_IV_LOGICAL_TABLES

Column Name	Datatype	Description
IS_VALID	VARCHAR2(13)	Is valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–163 ALL_IV_LOGICAL_TABLE_FK_USAGES

Column Name	Datatype	Description
LOGICAL_TABLE_ID	NUMBER(9)	ID of the logical table
LOGICAL_TABLE_NAME	VARCHAR2(1000)	Name of the logical table
FOREIGN_KEY_ID	NUMBER(9)	Foreign key ID
FOREIGN_KEY_NAME	VARCHAR2(1000)	Name of the foreign key
LOGICAL_TABLE_ID	NUMBER(9)	Logical table ID
LOGICAL_TABLE_NAME	VARCHAR2(1000)	Name of the logical table

Table 2–164 ALL_IV_LOGICAL_TABLE_AGGLEVELS

Column Name	Datatype	Description
LOGICAL_TABLE_ID	NUMBER(9)	Logical table ID
LOGICAL_TABLE_NAME	VARCHAR2(1000)	Name of the logical table
DRILL_PATH_ID	NUMBER(9)	Drill path ID
DRILL_PATH_NAME	VARCHAR2(1000)	Name of the drill path
DRILL_LEVEL_ID	NUMBER(9)	Drill level ID
DRILL_LEVEL_NAME	VARCHAR2(1000)	Name of the drill level

2.16 Real Time Views

Table 2–165 ALL_IV_STREAMS_QUEUES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
QUEUE_ID	NUMBER(9)	ID of the queue
QUEUE_NAME	VARCHAR2(255)	Name of the queue
BUSINESS_NAME	VARCHAR2(1000)	Business name of the queue
DESCRIPTION	VARCHAR2(4000)	Description of the queue
QUEUE_TABLE	VARCHAR2(255)	Physical name of the queue table
QUEUE_TABLE_ID	NUMBER(9)	ID of the queue table
PAYLOAD_TYPE	CHAR(11)	Type of payload
IS_VALID	VARCHAR2(13)	Is the queue valid

Table 2–165 (Cont.) ALL_IV_STREAMS_QUEUES

Column Name	Data Type	Description
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–166 ALL_IV_QUEUES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
QUEUE_ID	NUMBER(9)	ID of the queue
QUEUE_NAME	VARCHAR2(255)	Name of the queue
BUSINESS_NAME	VARCHAR2(1000)	Business name of the queue
DESCRIPTION	VARCHAR2(4000)	Description of the queue
QUEUE_TABLE	VARCHAR2(255)	Physical name of the queue table
QUEUE_TABLE_ID	NUMBER(9)	ID of the queue table
PAYLOAD_TYPE	VARCHAR2(255)	Type of payload for the queue
PAYLOAD_TYPE_ID	NUMBER	ID of the payload type
IS_VALID	VARCHAR2(13)	Is the queue valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–167 ALL_IV_QUEUE_PROPAGATIONS

Column Name	Data Type	Description
QUEUEPROPAGATION_ID	NUMBER(9)	ID of the queue propagation
QUEUETABLE_NAME	VARCHAR2(255)	Physical name of the queue table
BUSINESS_NAME	VARCHAR2(1000)	Business name of the queue table
DESCRIPTION	VARCHAR2(4000)	Description of the queue table
SOURCE_QUEUE	VARCHAR2(255)	Name of the source queue
SOURCE_QUEUE_ID	NUMBER(9)	ID of the source queue
TARGET_QUEUE	VARCHAR2(255)	Name of the target queue
TARGET_QUEUE_ID	NUMBER(9)	ID of the target queue
IS_VALID	VARCHAR2(13)	Is the queue table valid
UPDATED_ON	DATE	Update timestamp

Table 2–167 (Cont.) ALL_IV_QUEUE_PROPAGATIONS

Column Name	Data Type	Description
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–168 ALL_IV_QUEUE_TABLES

Column Name	Data Type	Description
SCHEMA_ID	NUMBER(9)	ID of the schema
SCHEMA_NAME	VARCHAR2(255)	Name of the schema
QUEUETABLE_ID	NUMBER(9)	ID of the queue table
QUEUETABLE_NAME	VARCHAR2(255)	Name of the queue table
BUSINESS_NAME	VARCHAR2(1000)	Business name of the queue table
DESCRIPTION	VARCHAR2(4000)	Description of the queue table
PAYLOAD_TYPE	VARCHAR2(767)	Type of payload
PAYLOAD_TYPE_ID	NUMBER(9)	ID of payload type
IS_VALID	VARCHAR2(13)	Is queue table valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–169 ALL_IV_STREAMS_CAPTURE

Column Name	Data Type	Description
STREAMSCAPTURE_ID	NUMBER(9)	ID of the streams capture
STREAMSCAPTURE_NAME	VARCHAR2(255)	Name of the streams capture
BUSINESS_NAME	VARCHAR2(1000)	Business name of the streams capture
DESCRIPTION	VARCHAR2(4000)	Description of the streams capture
STREAMS_QUEUE	VARCHAR2(255)	Streams queue
STREAMS_QUEUE_ID	NUMBER(9)	ID of the streams queue
IS_VALID	VARCHAR2(13)	Validation status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–170 ALL_IV_CAPTURE_RELATIONS

Column Name	Data Type	Description
STREAMS_CAPTURE_ID	NUMBER(9)	ID of the streams capture
STREAMS_CAPTURE_NAME	VARCHAR2(255)	Name of the streams capture
TABLE_ID	NUMBER(9)	ID of the table
TABLE_NAME	VARCHAR2(255)	Name of the table
CAPTURERELATION_ID	NUMBER(9)	ID of the capture relation
CAPTUREREALTION_NAME	VARCHAR2(255)	Name of the capture relation
BUSINESS_NAME	VARCHAR2(1000)	Business name of the capture relation
DESCRIPTION	VARCHAR2(4000)	Description of the capture relation
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.17 Scheduling Views

Table 2–171 ALL_IV_SCHEDULABLE

Column Name	Data Type	Description
OBJECT_ID	NUMBER(9)	ID of the object
OBJECT_TYPE	VARCHAR2(4000)	Type of the object
OBJECT_NAME	VARCHAR2(255)	Name of the object
CONFIGURATION_ID	NUMBER(9)	ID of the configuration
CONFIGURATION_ NAME	VARCHAR2(255)	Name of the configuration
SCHEDULE_ID	NUMBER(9)	ID of the applied schedule
SCHEDULE_NAME	VARCHAR2(255)	Name of the applied schedule

Table 2–172 ALL_IV_CALENDAR_SCHEDULES

Column Name	Data Type	Description
CALENDAR_ID	NUMBER(9)	ID of the calendar
CALENDAR_NAME	VARCHAR2(255)	Name of the calendar
SCHEDULE_ID	NUMBER(9)	ID of the schedule
SCHEDULE_NAME	VARCHAR2(255)	Name of the schedule
BUSINESS_NAME	VARCHAR2(1000)	Business name of the schedule
DESCRIPTION	VARCHAR2(4000)	Description of the schedule
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–172 (Cont.) ALL_IV_CALENDAR_SCHEDULES

Column Name	Data Type	Description
STARTTIME	DATE	Start time of the schedule
ENDTIME	DATE	End time of the schedule
REPEATEXPRESSION	VARCHAR2(4000)	Expression defining how often the schedule is active
TIMEZONE	VARCHAR2(255)	Time zone that the start and end time refer to

2.18 Security Views

Table 2–173 WBA_IV_OBJ_PRIVS

Column Name	Data Type	Description
GRANTEE	VARCHAR2(255)	Grantee name, user, or role receiving the grant
OBJECT_ID	NUMBER(9)	ID of the object
OBJECT_NAME	VARCHAR2(255)	Name of the object
OBJECT_TYPE	VARCHAR2(4000)	Type of object
PRIVILEGE	VARCHAR2(100)	Object privilege granted to the grantee

Table 2–174 WBA_IV_ROLE_PRIVS

Column Name	Data Type	Description
GRANTEE	VARCHAR2(255)	Grantee name, user, or role receiving the grant
GRANTED_ROLE	VARCHAR2(255)	The granted role

Table 2–175 WBA_IV_SYS_PRIVS

Column Name	Data Type	Description
GRANTEE	VARCHAR2(255)	Grantee name, user, or role receiving the grant
PRIVILEGE	VARCHAR2(100)	System privilege name granted to the grantee

Table 2–176 WBA_IV_ROLES

Column Name	Data Type	Description
ROLE_ID	NUMBER(9)	ID of the role
ROLE_NAME	VARCHAR2(255)	Name of the role
BUSINESS_NAME	VARCHAR2(1000)	Business name of the role
DESCRIPTION	VARCHAR2(4000)	Description of the role
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–177 WBA_IV_USERS

Column Name	Data Type	Description
USER_ID	NUMBER(9)	ID of the user
USER_NAME	VARCHAR2(255)	Name of the user
BUSINESS_NAME	VARCHAR2(1000)	Business name of the user
DESCRIPTION	VARCHAR2(4000)	Description of the user
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.19 Code Template Views

Table 2–178 ALL_IV_CT_VARIABLES

Column Name	Datatype	Description
CT_ID	NUMBER(9)	ID of the code template
CT_NAME	VARCHAR2(1000)	Name of the code template
VARIABLE_ID	NUMBER(9)	Variable ID
VARIABLE_NAME	VARCHAR2(1000)	Variable name
BUSINESS_NAME	VARCHAR2(1000)	Business name of the variable
DESCRIPTION	VARCHAR2(4000)	Description of the variable
DATA_TYPE	VARCHAR2(40)	Data type of the variable
DEFAULT_VALUE	VARCHAR2(4000)	Default value of the variable
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–179 ALL_IV_CT_TASKS

Column Name	Datatype	Description
CT_ID	NUMBER(9)	ID of the code template
CT_NAME	VARCHAR2(1000)	Name of the code template
TASK_ID	NUMBER(9)	Task ID
TASK_NAME	VARCHAR2(1000)	Name of the task
BUSINESS_NAME	VARCHAR2(1000)	Business name of the task
DESCRIPTION	VARCHAR2(4000)	Description of the task
TASK_TYPE	VARCHAR2(40)	Type of task
BOUND_OBJECT_ID	NUMBER(9)	ID of the bound object
BOUND_OBJECT_NAME	VARCHAR2(40)	Name of the bound object

Table 2–179 (Cont.) ALL_IV_CT_TASKS

Column Name	Datatype	Description
MAIN	VARCHAR2(4000)	Main
PREPROCESSING	VARCHAR2(4000)	Preprocessing
POSTPROCESSING	VARCHAR2(4000)	Postprocessing
INSTRUCTION	VARCHAR2(4000)	Instruction
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–180 ALL_IV_CT_FOLDERS

Column Name	Datatype	Description
PROJECT_ID	NUMBER(9)	Project ID
PROJECT_NAME	VARCHAR2(1000)	Name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	Information system ID
INFORMATION_SYSTEM_NAME	VARCHAR2(1000)	Name of the information system
BUSINESS_NAME	VARCHAR2(1000)	Business name of the folder
DESCRIPTION	VARCHAR2(4000)	Description of the folder
VALID	CHAR(1)	Flag to indicate if valid
IS_VALID	VARCHAR2(13)	Is valid
STATUS	VARCHAR2(40)	Status
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–181 ALL_IV_CT_TRANSITIONS

Column Name	Datatype	Description
CT_ID	NUMBER(9)	ID of the code template
CT_NAME	VARCHAR2(1000)	Name of the code template
TRANSITION_ID	NUMBER(9)	Transition ID
TRANSITION_NAME	VARCHAR2(1000)	Name of the transition
BUSINESS_NAME	VARCHAR2(1000)	Business name of the transition
DESCRIPTION	VARCHAR2(4000)	Description of the transition
CONDITION	VARCHAR2(4000)	Condition
TRANSITION_ORDER	NUMBER(9)	Transition order
SOURCE_ACTIVITY_ID	NUMBER(9)	ID of the source activity
SOURCE_ACTIVITY_NAME	VARCHAR2(1000)	Name of the source activity

Table 2–181 (Cont.) ALL_IV_CT_TRANSITIONS

Column Name	Datatype	Description
TARGET_ACTIVITY_ID	NUMBER(9)	ID of the target activity
TARGET_ACTIVITY_NAME	VARCHAR2(1000)	Name of the target activity
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–182 ALL_IV_CT_PARAMETERS

Column Name	Datatype	Description
PARAMETER_OWNER_ID	NUMBER(9)	ID of the owner
PARAMETER_OWNER_NAME	VARCHAR2(1000)	Name of the owner
PARAMETER_OWNER_TYPE	VARCHAR2(4)	Type of parameter owner
PARAMETER_ID	NUMBER(9)	ID of the parameter
PARAMETER_NAME	VARCHAR2(1000)	Name of the parameter
BUSINESS_NAME	VARCHAR2(1000)	Business name of the parameter
DESCRIPTION	VARCHAR2(4000)	Description of the parameter
POSITION	NUMBER(9)	Position
DATA_TYPE	VARCHAR2(40)	Data type of the parameter
DEFAULT_VALUE	VARCHAR2(4000)	Default value of the parameter
DIRECTION	VARCHAR2(5)	Direction
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–183 ALL_IV_CTS

Column Name	Datatype	Description
MODULE_ID	NUMBER(9)	Module ID
MODULE_NAME	VARCHAR2(1000)	Name of the module
CT_ID	NUMBER(9)	ID of the code template
CT_NAME	VARCHAR2(1000)	Name of the code template
BUSINESS_NAME	VARCHAR2(1000)	Business name of the code template
DESCRIPTION	VARCHAR2(4000)	Description of the code template
CT_TYPE	VARCHAR2(13)	Type of code template
BOUND_OBJECT_ID	NUMBER(9)	ID of the bound object
BOUND_OBJECT_NAME	VARCHAR2(40)	Name of the bound object
VALID	CHAR(1)	Flag to indicate valid

Table 2–183 (Cont.) ALL_IV_CTS

Column Name	Datatype	Description
IS_VALID	VARCHAR2(13)	Is valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–184 ALL_IV_EXEC_UNIT_MAP_COMP_USES

Column Name	Data Type	Description
EXECUTION_UNIT_ID	NUMBER(9)	ID of the execution unit
EXECUTION_UNIT_NAME	VARCHAR2(1000)	Name of the execution unit
MAP_COMPONENT_ID	NUMBER(9)	ID of the map component
MAP_COMPONENT_NAME	VARCHAR2(1000)	Name of the map component

Table 2–185 ALL_IV_EXEC_UNITS

Column Name	Data Type	Description
MAP_ID	NUMBER(9)	ID of the map
MAP_NAME	VARCHAR2(1000)	Name of the map
EXECUTION_UNIT_ID	NUMBER(9)	ID of the execution unit
EXECUTION_UNIT_NAME	VARCHAR2(1000)	Name of the execution unit
BUSINESS_NAME	VARCHAR2(1000)	Business name
DESCRIPTION	VARCHAR2(4000)	Desciption
CONFIGURATION_ID	NUMBER(9)	ID of the configuration
CONFIGURATION_NAME	VARCHAR2(1000)	Name of the configuration
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–186 ALL_IV_EXEC_UNIT_CT_USES

Column Name	Data Type	Description
EXECUTION_UNIT_ID	NUMBER(9)	ID of the execution unit
EXECUTION_UNIT_NAME	VARCHAR2(1000)	Name of the execution unit
CODE_TEMPLATE_TYPE	VARCHAR2(22)	Type of CT
CODE_TEMPLATE_ID	NUMBER(9)	ID of CT
CODE_TEMPLATE_NAME	VARCHAR2(1000)	Name of the CT
PARAMETER_NAME	VARCHAR2(1000)	Parameter name
BUSINESS_NAME	VARCHAR2(4000)	Business name

Table 2–186 (Cont.) ALL_IV_EXEC_UNIT_CT_USES

Column Name	Data Type	Description
DESCRIPTION	VARCHAR2(4000)	Description
VALUE	VARCHAR2(4000)	Value
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.20 Web Services Views

Table 2–187 ALL_IV_APP_SERVER_MODULES

Column Name	Datatype	Description
PROJECT_ID	NUMBER (9)	ID of the project
PROJECT_NAME	VARCHAR2 (1000)	Name of the project
INFORMATION_SYSTEM_ID	NUMBER (9)	ID of the information system
INFORMATION_SYSTEM_NAME	VARCHAR2 (1000)	Name of the information system
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the application server module
DESCRIPTION	VARCHAR2 (4000)	Description of the application server module
DATABASE_LINK	VARCHAR2 (40)	Database link used by the application server module.
VALID	CHAR (1)	The validation result of the application server module. 1 indicates valid and 0 indicates invalid.
IS_VALID	VARCHAR2 (13)	String to represent the validation value; can be Y, N, or Not Validated.
STATUS	VARCHAR2 (40)	The module status; is always set to Development for an application server module.
LOCATION_ID	NUMBER (9)	ID of the location
LOCATION_NAME	VARCHAR2 (1000)	Name of the location
LOCATION_TYPE	VARCHAR2 (40)	Type of location
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–188 ALL_IV_WEB_SERVICE_PACKAGES

Column Name	Datatype	Description
SCHEMA_ID	NUMBER (9)	ID of the schema
SCHEMA_NAME	VARCHAR2 (1000)	Name of the schema
PACKAGE_ID	NUMBER (9)	Package ID
PACKAGE_NAME	VARCHAR2 (1000)	Name of the package

Table 2–188 (Cont.) ALL_IV_WEB_SERVICE_PACKAGES

Column Name	Datatype	Description
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the package
DESCRIPTION	VARCHAR2 (4000)	Description of the package
VALID	CHAR (1)	The validation result of the web service package. 1 indicates valid and 0 indicates invalid.
IS_VALID	VARCHAR2 (13)	String to represent the validation value; can be Y, N, or Not Validated.
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–189 ALL_IV_WEB_SERVICES

Column Name	Datatype	Description
SCHEMA_ID	NUMBER (9)	ID of the schema
SCHEMA_NAME	VARCHAR2 (1000)	Name of the schema
PACKAGE_ID	NUMBER	Package ID
PACKAGE_NAME	VARCHAR2 (1000)	Package name
WEBSERVICE_ID	NUMBER (9)	ID of the webservice
WEBSERVICE_NAME	VARCHAR2 (1000)	Name of the webservice
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the webservice
DESCRIPTION	VARCHAR2 (4000)	Description of the webservice
ASYNC	CHAR (1)	Indicates whether the web service is asynchronous or not.
SOAP_VERSION	VARCHAR2 (40)	SOAP version
WSDL_URL	VARCHAR2 (4000)	URL of WSDL
WSDL_VERSION	VARCHAR2 (40)	WSDL version
VALID	CHAR (1)	The validation result of the web service. 1 indicates Valid and 0 indicates Invalid.
IS_VALID	VARCHAR2 (13)	String to represent the validation value; can be Y, N, or Not Validated.
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

2.21 Others

Table 2–190 ALL_IV_ACTIVITY_FOLDERS

Column Name	Data Type	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(255)	Name of the project
ACTIVITY_FOLDER_ID	NUMBER(9)	ID of the activity folder
ACTIVITY_FOLDER_NAME	VARCHAR2(255)	Name of the activity folder
BUSINESS_NAME	VARCHAR2(1000)	Business name of the activity folder
DESCRIPTION	VARCHAR2(4000)	Description of the activity folder
IS_VALID	VARCHAR2(13)	Is the activity folder valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–191 ALL_IV_ACTIVITY_TEMPLATES

Column Name	Data Type	Description
ACTIVITY_FOLDER_ID	NUMBER(9)	ID of the activity folder
ACTIVITY_FOLDER_NAME	VARCHAR2(255)	Name of the activity folder
ACTIVITY_TEMPLATE_ID	NUMBER(9)	ID of the activity template
ACTIVITY_TEMPLATE_NAME	VARCHAR2(255)	Name of the activity template
BUSINESS_NAME	VARCHAR2(1000)	Business name of the activity template
DESCRIPTION	VARCHAR2(4000)	Description of the activity template
IS_VALID	VARCHAR2(13)	Is the activity template valid
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–192 ALL_IV_PLS_COLLECTIONS

Column Name	Data Type	Description
LIBRARY_ID	NUMBER(9)	ID of the library
LIBRARY_NAME	VARCHAR2(255)	Name of the library
COLLECTION_ID	NUMBER(9)	ID of the collection
COLLECTION_NAME	VARCHAR2(255)	Name of the collection
BUSINESS_NAME	VARCHAR2(1000)	Business name of the collection
DESCRIPTION	VARCHAR2(4000)	Description of the collection
COLLECTION_TYPE	VARCHAR2(255)	Type of the collection

Table 2–192 (Cont.) ALL_IV_PLS_COLLECTIONS

Column Name	Data Type	Description
RELATED_RECORD_ID	NUMBER(9)	ID of the related PLS record
RELATED_RECORD_NAME	VARCHAR2(255)	Name of the related PLS record
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–193 ALL_IV_PLS_RECORDS

Column Name	Data Type	Description
LIBRARY_ID	NUMBER(9)	ID of the library
LIBRARY_NAME	VARCHAR2(255)	Name of the library
RECORD_ID	NUMBER(9)	ID of the record
RECORD_NAME	VARCHAR2(255)	Name of the record
BUSINESS_NAME	VARCHAR2(1000)	Business name of the record
DESCRIPTION	VARCHAR2(4000)	Description of the record
RECORD_TYPE	VARCHAR2(40)	Type of the record
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–194 ALL_IV_ROW_RELATIONSHIPS

Column Name	Data Type	Description
PROFILE_ID	NUMBER(9)	ID of the profile
PROFILE_NAME	VARCHAR2(255)	Name of the profile
ENTITY_ID	NUMBER(9)	ID of the entity
ENTITY_NAME	VARCHAR2(255)	Name of the entity
ROW_RELATIONSHIP_ID	NUMBER(9)	ID of the row relationship
ROW_RELATIONSHIP_NAME	VARCHAR2(255)	Name of the row relationship
BUSINESS_NAME	VARCHAR2(1000)	Business name of the row relationship
REMOTE_KEY_ID	NUMBER(9)	ID of the other row relationship
IS_DISCOVERED	CHAR(3)	If this row relationship was discovered
IS_DOCUMENTED	CHAR(2)	If this row relationship is documented
LOCAL_MAX_CARDINALITY	VARCHAR2(40)	Maximum number of values found on the local side
LOCAL_MIN_CARDINALITY	VARCHAR2(40)	Minimum number of values found on the local side
REMOTE_MAX_CARDINALITY	VARCHAR2(40)	Maximum number of values found on the remote side

Table 2–194 (Cont.) ALL_IV_ROW_RELATIONSHIPS

Column Name	Data Type	Description
REMOTE_MIN_CARDINALITY	VARCHAR2(40)	Minimum number of values found on the remote side
NUM_ORPHANS	VARCHAR2(40)	Number of distinct values found in the local column but not in the remote column
COMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that satisfy the discovered row relationship
COMPLIANT_CNT_QUERY	VARCHAR2(4000)	Not used
NONCOMPLIANT_QUERY	VARCHAR2(4000)	The query used to get the rows that do not satisfy the discovered common format
NONCOMPLIANT_CNT_QUERY	VARCHAR2(4000)	Not used
DRILLDOWN_QUERY	VARCHAR2(4000)	The query used to get all the rows of the table but distinguishes which row satisfies the discovered row relationship and which does not
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–195 SUB_FQ_PUSAGE_DEF

Column Name	Data Type	Description
CLASSTYPENAME	VARCHAR2(255)	Object type name
PUSAGEID	NUMBER(9)	ID of the usage definition of the property. (The attributes of a property may be context specific. For example, columns of an external table may have different properties, or the properties may have different constraints.)
PDEFID	NUMBER(9)	ID of the property definition
PROPNAME	VARCHAR2(255)	The name of the property as used in OMB*Plus
PUSAGETYPE	VARCHAR2(7)	Property type for this usage, which determines where it is automatically displayed in the UI: CORE: Not displayed LOGICAL: Editor Property Inspector CONFIG: Configuration Property Inspector USERDEF: User Defined Properties tab
STORAGETYPE	VARCHAR2(7)	Storage type for this property, which determines how it is physically stored: CORE: Column of object row LOGICAL: Child row of object CONFIG: Child row of configuration of object USERDEF: Child row of object
DATATYPE	NUMBER	ID for the data type of this usage
TYPENAME	VARCHAR2(255)	Data type name of this usage

Table 2–195 (Cont.) SUB_FQ_PUSAGE_DEF

Column Name	Data Type	Description
HIDDEN	VARCHAR2(4000)	Is this usage shown in the UI (PropertyInspector)? Value is true or false
PUBLISHTOPUBLICAPI	VARCHAR2(4000)	Is this usage included in OMB*Plus scripting documentation? Value is true or false
CONTRIBUTETOSIGNATURE	VARCHAR2(4000)	Do changes in this property affect the semantic value of the owning object with respect to mapping or deployment? That is, would a change in the value of this property require the owning object to be regenerated before being deployed. Value is true or false
DYNAMICDEFAULT	VARCHAR2(4000)	Fully qualified name of the Java class providing the value for this property if it has not been set. This is called each time the property value is requested. (If this value is defined, then the value shown in the DEFAULTVALUE column should be considered invalid.)
DEFAULTVALUE	VARCHAR2(4000)	The static, default value for the usage. The primary default value of usage, property, domain, and primitive type, in that order. DYNAMIC DEFAULT takes precedence if defined. (Note that Null is not used as an indicator for this because Null is a potential legitimate value, and is specifically the default for any reference property)
PSETID	NUMBER(9)	The ID of the property set containing this usage. (Property sets are the context grouping mechanism for usage definitions. Usage type is actually defined at the property set level.)
PSETNAME	VARCHAR2(255)	The name of the property set containing this usage. (Also referred to as the selector key – generally gives indication of the context in which this property set is active. For example, maps have different sets based on generation language with the names SQLLOADER, PLSQL, ABAP, and so on. The value DEFAULT indicates the name selector mechanism is not used for this property set.)
PSETCLASSNAME	VARCHAR2(255)	Type name of the parent context for this property set. If defined, this property set is only active when the owning object is a child of an object of this type. For example, Index has different configuration property sets for the child object of Table or Materialized View
NLSKEY	VARCHAR2(255)	NLS lookup key base for localized property name, description
DESCRIPTION	VARCHAR2(4000)	Description of the usage from the definition file. Usually the same as the default <NLSKEY>:DESCRIPTION (English) shown in the UI/OMB*Plus.
GROUPID	NUMBER(9)	The ID of the display group for this property. Used only in the UI (Property Inspector category)
GROUPNAME	VARCHAR2(255)	Name of the group from the definition file. Usually in uppercase, with the spaces converted to underscore. Version of the default <GROUPDEFINITIONKEY>:NAME (English) shown in the property inspector
GROUPDEFINITIONKEY	VARCHAR2(255)	The NLS lookup key base for localized group name

Table 2–195 (Cont.) SUB_FQ_PUSAGE_DEF

Column Name	Data Type	Description
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–196 ALL_IV_FLD_CONTAINED_OBJECTS

Column Name	Data Type	Description
FOLDER_ID	NUMBER (9)	ID of the folder
FOLDER_NAME	VARCHAR2(1000)	Name of the folder
CONTAINED_OBJECT_ID	NUMBER(9)	ID of the object contained in the folder
CONTAINED_OBJECT_NAME	VARCHAR2(1000)	Name of the object contained in the folder
DESCRIPTION	VARCHAR2(4000)	Description
BUSINESS_NAME	VARCHAR2(1000)	Business name
CONTAINED_OBJECT_PATH	VARCHAR2(4000)	Path of the contained object
REFERENCED_OBJECT_TYPE	VARCHAR2(4000)	Type of the referenced object
REFERENCED_OBJECT_ID	NUMBER(9)	ID of the referenced object
REFERENCED_OBJECT_NAME	VARCHAR2(1000)	Name of the referenced object
CONTAINED	CHAR(1)	Flag to indicate if it is contained
IS_CONTAINED	VARCHAR2(7)	Is contained
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–197 ALL_IV_FOLDERS

Column Name	Data Type	Description
OWNER_ID	NUMBER(9)	ID of the owner
OWNER_NAME	VARCHAR2(1000)	Name of the owner
FOLDER_ID	NUMBER(9)	ID of the folder
FOLDER_NAME	VARCHAR2(1000)	Name of the folder
BUSINESS_NAME	VARCHAR2(1000)	Business Name
DESCRIPTION	VARCHAR2(4000)	Description
VALID	CHAR(1)	Flag to indicate if it is valid
IS_VALID	VARCHAR2(13)	Is valid
UPDATED_ON	DATE	Update timestamp

Table 2–197 (Cont.) ALL_IV_FOLDERS

Column Name	Data Type	Description
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–198 ALL_IV_DATABASE_MODULES

Column Name	Datatype	Description
PROJECT_ID	NUMBER(9)	ID of the project
PROJECT_NAME	VARCHAR2(1000)	Name of the project
INFORMATION_SYSTEM_ID	NUMBER(9)	ID of the information system
INFORMATION_SYSTEM_NAME	VARCHAR2(1000)	Name of the information system
BUSINESS_NAME	VARCHAR2(1000)	Business name
DESCRIPTION	VARCHAR2(4000)	Description
DATABASE_LINK	VARCHAR2(40)	Database link
STRONG_TYPE_NAME	VARCHAR2(255)	Strong type name
PLATFORM_ID	NUMBER(9)	Platform ID
PLATFORM_NAME	VARCHAR2(1000)	Platform Name
VALID	CHAR(1)	Flag to indicate if it is valid
IS_VALID	VARCHAR2(13)	Is valid
STATUS	VARCHAR2(40)	Status
LOCATION_ID	NUMBER(9)	Location ID
LOCATION_NAME	VARCHAR2(1000)	Location Name
METADATA_LOCATION_ID	NUMBER(9)	ID of the Metadata Location
METADATA_LOCATION_NAME	VARCHAR2(1000)	Name of the Metadata Location
MODULE_TYPE	VARCHAR2(1000)	Module Type
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–199 ALL_IV_CLASS_DEFINITIONS

Column Name	Datatype	Description
CLASS_DEFINITION_ID	NUMBER (9)	ID of the class definition
CLASS_DEFINITION_NAME	VARCHAR2 (1000)	Name of the class definition
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the class definition
DESCRIPTION	VARCHAR2 (4000)	Description of the class definition
OWNING_MODEL	VARCHAR2 (1000)	Owning model
OWNING_MODEL_ID	NUMBER (9)	Owning model ID

Table 2–199 (Cont.) ALL_IV_CLASS_DEFINITIONS

Column Name	Datatype	Description
SCRIPTING_NAME	VARCHAR2 (1000)	Scripting name
ABSTRACT	CHAR (1)	Flag
CLASS_TYPE	VARCHAR2 (20)	Class type
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–200 ALL_IV_ATTRIBUTES

Column Name	Datatype	Description
SCHEMA_ID	NUMBER (9)	ID of the schema
SCHEMA_NAME	VARCHAR2 (1000)	Schema name
ENTITY_ID	NUMBER (9)	ID of the entity
ENTITY_TYPE	VARCHAR2 (4000)	Entity type
ENTITY_NAME	VARCHAR2 (1000)	Entity name
ATTRIBUTE_SET_NAME	VARCHAR2 (1000)	Name of the attribute set
ATTRIBUTE_SET_ID	NUMBER (9)	Attribute set ID
ATTRIBUTE_NAME	VARCHAR2 (1000)	Name of the attribute
POSITION	NUMBER (9)	Position of the attribute
ATTRIBUTE_ID	NUMBER (9)	Attribute ID
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the attribute
DESCRIPTION	VARCHAR2 (4000)	Description of the attribute
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–201 ALL_IV_ATTRIBUTE_SETS

Column Name	Datatype	Description
SCHEMA_ID	NUMBER (9)	Schema ID
SCHEMA_NAME	VARCHAR2 (1000)	Name of the schema
ENTITY_ID	NUMBER (9)	Entity ID
ENTITY_TYPE	VARCHAR2 (4000)	Entity type
ENTITY_NAME	VARCHAR2 (1000)	Name of the entity
ATTRIBUTE_SET_NAME	VARCHAR2 (1000)	Name of the attribute set
ATTRIBUTE_SET_ID	NUMBER (9)	Attribute set ID
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the attribute set

Table 2–201 (Cont.) ALL_IV_ATTRIBUTE_SETS

Column Name	Datatype	Description
DESCRIPTION	VARCHAR2 (4000)	Description of the attribute set
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–202 ALL_IV_XFORM_MAP_GROUPS

Column Name	Datatype	Description
MAP_ID	NUMBER (9)	ID of the map
MAP_NAME	VARCHAR2 (1000)	Name of the map
MAP_COMPONENT_ID	NUMBER (9)	ID of the map component
MAP_COMPONENT_NAME	VARCHAR2 (1000)	Name of the map component
PARAMETER_GROUP_ID	NUMBER (9)	ID of the parameter group
PARAMETER_GROUP_NAME	VARCHAR2 (1000)	Name of the parameter group
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the map
PARAMETER_GROUP_TYPE	VARCHAR2 (5)	Parameter group type
DESCRIPTION	VARCHAR2 (4000)	Description of the Map
POSITION	NUMBER (9)	Position of the map
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–203 ALL_IV_WORKSPACE_ASSIGNMENTS

Column Name	Datatype	Description
WORKSPACE_ID	NUMBER (9)	ID of the workspace
WORKSPACE_NAME	VARCHAR2 (1000)	Name of the workspace
WORKSPACE_BUSINESS_NAME	VARCHAR2 (1000)	Business name of the workspace
WORKSPACE_DESCRIPTION	VARCHAR2 (4000)	Description of the workspace
USER_NAME	VARCHAR2 (1000)	User name
ISWORKSPACEOWNER	CHAR (1)	Flag to indicate if the user is workspace owner
USER_CREATEDBY	VARCHAR2 (40)	User created by
USER_CREATIONTIMESTAMP	DATE	Timestamp when user is created
USER_UPDATEBY	VARCHAR2 (40)	Updated by
USER_UPDATETIMESTAMP	DATE	Update timestamp

Table 2–204 ALL_IV_WORKSPACES

Column Name	Datatype	Description
WORKSPACE_ID	NUMBER (9)	Workspace ID
WORKSPACE_NAME	VARCHAR2 (1000)	Workspace name
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the workspace
DESCRIPTION	VARCHAR2 (4000)	Description of the workspace
WORKSPACE_OWNER	VARCHAR2 (1000)	Owner of the workspace
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–205 ALL_IV_SUBPARTITION_TEMPLATES

Column Name	Datatype	Description
SCHEMA_ID	NUMBER (9)	Schema ID
SCHEMA_NAME	VARCHAR2 (1000)	Name of the schema
ENTITY_ID	NUMBER (9)	Entity ID
ENTITY_TYPE	VARCHAR2 (4000)	Type of entity
ENTITY_NAME	VARCHAR2 (1000)	Name of the entity
PARTITION_ID	NUMBER (9)	Partition ID
PARTITIONING_TYPE	VARCHAR2 (40)	Type of partition
NAME	VARCHAR2 (1000)	Name
BUSINESS_NAME	VARCHAR2 (1000)	Business name
DESCRIPTION	VARCHAR2 (4000)	Description
POSITION	VARCHAR2 (40)	Position of partition
VALUE_CLAUSE	VARCHAR2 (4000)	Value clause
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–206 ALL_IV_SUBPARTITIONS

Column Name	Datatype	Description
ENTITY_ID	NUMBER (9)	Entity ID
ENTITY_TYPE	VARCHAR2 (4000)	Type of entity
ENTITY_NAME	VARCHAR2 (1000)	Name of the entity
PARENT_PARTITION_ID	NUMBER (9)	ID of the parent partition
PARENT_NAME	VARCHAR2 (1000)	Name of the parent
PARTITIONING_TYPE	VARCHAR2 (40)	Type of partitioning

Table 2–206 (Cont.) ALL_IV_SUBPARTITIONS

Column Name	Datatype	Description
SUBPARTITION_ID	NUMBER (9)	ID of the subpartition
NAME	VARCHAR2 (1000)	Name of the subpartition
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the subpartition
DESCRIPTION	VARCHAR2 (4000)	Description of the subpartition
POSITION	VARCHAR2 (40)	Position of the subpartition
VALUE_CLAUSE	VARCHAR2 (4000)	Value clause
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–207 ALL_IV_SUBPARTITION_KEYS

Column Name	Datatype	Description
SCHEMA_ID	NUMBER (9)	Schema ID
SCHEMA_NAME	VARCHAR2 (1000)	Name of the schema
ENTITY_ID	NUMBER (9)	Entity ID
ENTITY_TYPE	VARCHAR2 (4000)	Type of entity
ENTITY_NAME	VARCHAR2 (1000)	Name of the entity
PARTITIONING_TYPE	VARCHAR2 (40)	Partition type
SUBPARTITION_KEY_ COLUMN_ID	NUMBER (9)	ID of the subpartition key column
SUBPARTITION_KEY_COLUMN	VARCHAR2 (1000)	Name
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the subpartition key
DESCRIPTION	VARCHAR2 (4000)	Description of the subpartition key
POSITION	NUMBER (9)	Position of the subpartition key
COUNT	NUMBER (9)	Count
HASH_PARTITION_QUANTITY	VARCHAR2 (40)	Hash partition quantity
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–208 ALL_IV_PROPERTY_DEFINITIONS

Column Name	Datatype	Description
PROPERTY_DEFINITION_ID	NUMBER (9)	Property definition ID
PROPERTY_DEFINITION_NAME	VARCHAR2 (1000)	Name of the property definition
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the property definition

Table 2–208 (Cont.) ALL_IV_PROPERTY_DEFINITIONS

Column Name	Datatype	Description
PREDEFINED	CHAR (1)	Flag to indicate if its predefined.
DESCRIPTION	VARCHAR2 (4000)	Description of the property definition
OWNING_CLASS	VARCHAR2 (1000)	Owning class
OWNING_CLASS_ID	NUMBER (9)	Owning class ID
OWNING_PROPERTY_SET_ID	NUMBER (9)	ID of the owning property set
DEFAULT_VALUE	VARCHAR2 (4000)	Default value
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–209 ALL_IV_PARTITIONS

Column Name	Datatype	Description
SCHEMA_ID	NUMBER (9)	Schema ID
SCHEMA_NAME	VARCHAR2 (1000)	Name of the schema
ENTITY_ID	NUMBER (9)	Entity ID
ENTITY_TYPE	VARCHAR2 (4000)	Type of entity
ENTITY_NAME	VARCHAR2 (1000)	Name of the entity
PARTITION_ID	NUMBER (9)	Partition ID
PARTITION_TYPE	VARCHAR2 (40)	Type of partition
NAME	VARCHAR2 (1000)	Partition name
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the partition
DESCRIPTION	VARCHAR2 (4000)	Description of the partition
POSITION	VARCHAR2 (40)	Position of the partition
VALUE_CLAUSE	VARCHAR2 (4000)	Value clause
HASH_SUBPARTITION_COUNT	VARCHAR2 (40)	Hash subpartition count
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–210 ALL_IV_PLATFORMS

Column Name	Datatype	Description
PLATFORM_ID	NUMBER (9)	ID of the platform
PLATFORM_NAME	VARCHAR2 (1000)	Name of the platform
BUSINESS_NAME	VARCHAR2 (1000)	Business name of the platform
DESCRIPTION	VARCHAR2 (4000)	Description of the platform

Table 2–210 (Cont.) ALL_IV_PLATFORMS

Column Name	Datatype	Description
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–211 ALL_IV_PARTITION_KEYS

Column Name	Datatype	Description
SCHEMA_ID	NUMBER(9)	Schema ID
SCHEMA_NAME	VARCHAR2(1000)	Name of the schema
ENTITY_ID	NUMBER(9)	Entity ID
ENTITY_TYPE	VARCHAR2(4000)	Type of entity
ENTITY_NAME	VARCHAR2(1000)	Name of the entity
PARTITIONING_TYPE	VARCHAR2(40)	Partitioning type
PARTITION_KEY_COLUMN_ID	NUMBER(9)	ID of the partition key column
PARTITION_KEY_COLUMN	VARCHAR2(1000)	Partition key column
BUSINESS_NAME	VARCHAR2(1000)	Business name of the partition key
DESCRIPTION	VARCHAR2(4000)	Description of the partition key
POSITION	NUMBER(9)	Position
COUNT	NUMBER(9)	Count
HASH_PARTITION_QUANTITY	VARCHAR2(40)	Hash partition quantity
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–212 ALL_IV_INDEX_PARTITIONS

Column Name	Datatype	Description
SCHEMA_ID	NUMBER(9)	Schema ID
SCHEMA_NAME	VARCHAR2(1000)	Name of the schema
ENTITY_ID	NUMBER(9)	Entity ID
ENTITY_TYPE	VARCHAR2(4000)	Type of entity
ENTITY_NAME	VARCHAR2(1000)	Name of the entity
INDEX_ID	NUMBER(9)	ID of the index
INDEX_NAME	VARCHAR2(1000)	Name of the index
INDEX_TYPE	VARCHAR2(40)	Type of index
INDEX_PARTITIONING_TYPE	VARCHAR2(40)	Index partitioning type
PARTITION_ID	NUMBER(9)	Partition ID

Table 2–212 (Cont.) ALL_IV_INDEX_PARTITIONS

Column Name	Datatype	Description
NAME	VARCHAR2(1000)	Name of the partition
BUSINESS_NAME	VARCHAR2(1000)	Business name of the partition
DESCRIPTION	VARCHAR2(4000)	Description of the partition
POSITION	VARCHAR2(40)	Position
VALUE_CLAUSE	VARCHAR2(4000)	Value clause
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–213 ALL_IV_INDEX_PARTITION_KEYS

Column Name	Datatype	Description
SCHEMA_ID	NUMBER(9)	Schema ID
SCHEMA_NAME	VARCHAR2(1000)	Name of the schema
ENTITY_ID	NUMBER(9)	Entity ID
ENTITY_TYPE	VARCHAR2(4000)	Type of entity
ENTITY_NAME	VARCHAR2(1000)	Name of the entity
INDEX_NAME	VARCHAR2(1000)	Name of the index
INDEX_PARTITIONING_TYPE	VARCHAR2(40)	Index partitioning type
INDEX_ID	NUMBER(9)	Index ID
INDEX_PARTITION_KEY_ COLUMN_ID	NUMBER(9)	Column ID of the index partition key
INDEX_PARTITION_KEY_ COLUMN	VARCHAR2(1000)	Index partition key column
BUSINESS_NAME	VARCHAR2(1000)	Business name of the partition key
DESCRIPTION	VARCHAR2(4000)	Description of the partition key
POSITION	NUMBER(9)	Position
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Table 2–214 ALL_IV_INDEX_KEYS

Column Name	Datatype	Description
SCHEMA_ID	NUMBER(9)	Schema ID
SCHEMA_NAME	VARCHAR2(1000)	Name of the schema
ENTITY_ID	NUMBER(9)	Entity ID
ENTITY_TYPE	VARCHAR2(4000)	Type of entity

Table 2–214 (Cont.) ALL_IV_INDEX_KEYS

Column Name	Datatype	Description
ENTITY_NAME	VARCHAR2(1000)	Name of the entity
INDEX_NAME	VARCHAR2(1000)	Index name
INDEX_ID	NUMBER(9)	Index ID
INDEX_TYPE	VARCHAR2(40)	Type of index
INDEX_KEY_COLUMN_ID	NUMBER(9)	ID of the index key column
INDEX_KEY_COLUMN	VARCHAR2(1000)	Index key column
BUSINESS_NAME	VARCHAR2(1000)	Business name of the index key
DESCRIPTION	VARCHAR2(4000)	Description of the index key
POSITION	NUMBER(9)	Position of Index
UPDATED_ON	DATE	Update timestamp
CREATED_ON	DATE	Creation timestamp
UPDATED_BY	VARCHAR2(40)	Updated by user
CREATED_BY	VARCHAR2(40)	Created by user

Part II

OMB*Plus Scripting Language

This part contains the following chapters:

3

Introduction to OMB*Plus

OMB*Plus is a flexible, high-level command line metadata access tool for Oracle Warehouse Builder. Use OMB*Plus to create, modify, delete, and retrieve object metadata in Warehouse Builder design and runtime repositories.

This chapter contains the following topics:

3.1 About the OMB Scripting Language

The Warehouse Builder scripting language, known as OMB*Plus, is an extension of the Tcl programming language. With OMB*Plus, you can write the syntactic constructs such as variable support, conditional and looping control structures, error handling, and standard library procedures.

Use OMB*Plus to create, modify, delete, and retrieve object metadata in Warehouse Builder design and runtime repositories.

OMB*Plus enables you to edit Warehouse Builder repository metadata using a scripting interface. You can use this scripting interface to:

- Perform complex actions directly in Warehouse Builder, without launching the client user interface.
- Define sets of routine operations to be executed in Warehouse Builder.
- Perform batch operations in Warehouse Builder.
- Automate a series of conditional operations in Warehouse Builder.

3.1.1 Using OMB*Plus

To use OMB*Plus, first launch OMB*Plus and then connect to a repository. Type all commands and keywords in OMB*Plus using uppercase.

Whenever you switch between the GUI and OMB*Plus, ensure that you are working on the same project in both the environments. For example, if you switch projects in the Design Center without switching to the same project in the OMB*Plus environment, you may get unpredictable results while executing OMB*Plus commands. To switch projects in OMB*Plus, use OMBCC. To list the current project in the OMB*Plus environment, use OMBDCC.

3.1.1.1 Launching OMB*Plus

To launch the OMB*Plus console, follow the instructions specific to your operating system.

- **UNIX:** At the command prompt, enter:

  ```
  <OWB_HOME>/owb/bin/unix/OMBPlus.sh
  ```

- **Windows:** From **Start**, navigate to **Warehouse Builder**, located within Oracle-OUI. Select **OMB Plus** from the list of menu items.

  Alternatively, you can also launch OMB*Plus from the Warehouse Builder Design Center. To do this, from the main menu, select **View**, and then **OMB*Plus**.

3.1.1.2 Connecting to a Repository

From the OMB*Plus console, enter:

```
OMBCONNECT <repos>/<password>@<host>:<port>:<service_name> USE WORKSPACE
'<wokspace_name>'
```

In this expression, `<repos>` is the name of the design-time repository, `<host>` is the computer on which the repository is installed, `<servicename>` is the name of the database that contains the repository, and `<workspace_name>` is the name of the workspace. Note that the workspace name must be provided within single quotes (').

If the workspace and repository names are same, then you do not have to specify the workspace name. You can use the command:

OMBCONNECT <repos>/<password>@<host>:<port>:<service_name>

3.1.1.3 Getting Help for OMB*Plus Commands

Use the `OMBHELP` command to display help on Warehouse Builder commands. The help describes the purpose of the command, the syntax in BNF format, and each of the keywords or options.

For details and an example on how to use `OMBHELP`, see OMBHELP on page 3-9.

3.1.2 Writing OMB*Plus Commands

Keep in mind the following points when you execute OMB*Plus commands:

3.1.2.1 Specifying Values

You can set the value of a Boolean configuration property, for example the NOT_NULL property in the example, using any one of the following values: TRUE, FALSE, YES, NO, 1, or 0. When you set the value to TRUE, FALSE, YES, or NO, enclose the value in single quotes.

For example:

```
OMBCREATE TABLE 'EMP' \
ADD COLUMN 'EMPNO' SET PROPERTIES (NOT_NULL) VALUES (1)
```

Or

```
OMBCREATE TABLE 'EMP' \
ADD COLUMN 'EMPNO' SET PROPERTIES (NOT_NULL) VALUES ('true')
```

When you specify numeric values for a configuration property, do not enclose the values in single quotes.

3.1.2.2 Special Characters

Do not use a number sign (#) in the middle of an OMB*Plus command.

Do not use an exclamation point (!) because it is an illegal character.

Tcl interprets several special characters differently than Warehouse Builder and the Oracle database. Review this and subsequent sections to learn how to properly use the following characters in OMB*Plus: dollar sign ($), backslash (\), bracket ([ ), and semicolon (;).

Dollar sign ($): The dollar sign identifies Tcl variables in Tcl but has no special meaning in Warehouse Builder. Therefore, if you include $ in the name of a Warehouse Builder object, OMB*Plus uses the Tcl convention and displays the $ inside curly brackets such that *name$* displays as *name{$}*. This prevents the name from being misinterpreted as a variable.

Backslash (\): Tcl uses the backward slash to indicate the end of a line in a command that spans multiple lines. Therefore, in a multiple line OMB*Plus command, use only a backslash (\) after each line. This interpretation of the backslash has implications on how you specify full paths in OMB*Plus, as described in "Specifying Paths" on page 3-4.

Tcl also uses the backslash as the least preferred method for escaping special characters. Use curly braces as the preferred method as described in "Escaping Special Characters and Writing Complex Arguments" on page 3-3.

Semicolon(;): The semicolon separates two commands in Tcl. Using a semicolon in a quoted string results in an error. As a work around, escape the semicolon character by putting a backward slash (\) in front of the semicolon. For example,

```
OMBCREATE FLATFILE 'FF_DSR_RLE' \
SET PROPERTIES(DATA_FILE_NAME,IS_DEMILITED,CHARACTER_SET,RECORD_DELIMITER,\
FIELD_DELIMITER,FIELD_LEFT_ENCLOSURE,FIELD_RIGHT_ENCLOSURE) \
VALUES('DSR_RLE.dat','TRUE','WE8MSWIN1252','\n','\;','"','"')
```

3.1.2.3 Escaping Special Characters and Writing Complex Arguments

Tcl uses curly braces ({) and (}) as preferred method for escaping special characters and writing valid, complex arguments. For the following situations, you can enclose the element in braces and leave the contents alone:

- The element contains embedded spaces.
- The element contains one of the Special Characters on page 3-3.
- The element starts with a brace or double-quote.
- There are no characters in the element.

To escape single quotation marks in elements, consider using the procedure *OMBToSettableString* described in "Predefined Tcl Procedures" on page 3-5.

You should consider using the backslash for escaping only in the limited situations that using curly braces results in unmatched braces, the last character of the argument is a backslash, or the element contains a backslash followed by another backslash indicating a new line.

3.1.2.3.1 Escaping Special Characters in Passwords

Escaping special characters in passwords in OMB*Plus commands is affected both by Tcl special character handling and special character handling by the database. The affected commands include the following:

- OMBCONNECT
- OMBCONNECT CONTROL_CENTER
- OMBDEINSTALL OWB_RAC
- OMBDEINSTALL OWB_REPOSITORY
- OMBDEINSTALL OWB_TARGET_USER
- OMBEXPORT ENTIRE_REPOSITORY
- OMBGETLICENSE
- OMBIMPORT ENTIRE_REPOSITORY
- OMBINSTALL OWB_RAC
- OMBINSTALL OWB_REPOSITORY
- OMBINSTALL OWB_TARGET_USER
- OMBMLSUPDATE OWB_REPOSITORY
- OMBREGISTER USER
- OMBSEED
- OMBSEEDLICENSE
- OMBUPGRADEREPOSITORY

The rules for quoting are as follows:

- Because OMB scripting is TCL-based, if password contains: $, [,], or \, then enclose the whole connection string in TCL brace quotes.
- If the password contains special characters such as white space, @, or /, then the password (only) should be quoted using double-quotes.

Table 3–1 contains examples of correctly quoted password strings. In the examples, orcl is the name of the database that contains the repository and *rep_user* is the name of the repository user.

Table 3–1 Correctly Quoted Password Strings

Password	Connect String
[hello]	OMBCONN {rep_user/[hello]@localhost:1521:orcl}
	OMBCONN {rep_user/"[hello]"@localhost:1521:orcl}
hello world	OMBCONN rep_user/"hello world"@localhost:1521:orcl
@@@@@	OMBCONN rep_user/"@@@@@"@localhost:1521:orcl
/////	OMBCONN rep_user/"/////"@localhost:1521:orcl
\\\\\	OMBCONN {rep_user/\\\\\@localhost:1521:orcl}
	OMBCONN {rep_user/"\\\\\"@localhost:1521:orcl}

3.1.2.4 Specifying Paths

Do not use a backward slash (\) when you specify the full path for the commands that use the full path, such as, OMBIMPORT, OMBVALIDATE, OMBLOG, and so on. For example, in the following commands are invalid and the log file is not created:

```
set OMBLOG c:\my_project\omb_logfile.log (On Windows)
set OMBLOG \home\my_project\omb_logfile.log (On UNIX)
```

On UNIX, use a forward slash as the path separator. For example, the following command creates a log file.

```
set OMBLOG /home/my_project/omb_logfile.log
```

On Windows, you can use either a forward slash(/) or two backward slashes (\\) as a path separator. Alternately, you can use a backward slash in the path, but in this case, enclose the entire filename in curly braces. The following are examples of commands that you can use to create a log file.

```
set OMBLOG c:/my_project/omb_logfile.log
set OMBLOG c:\\my_project\\omb_logfile.log
set OMBLOG {c:\my_project\omb_logfile.log}
```

3.1.2.5 Predefined Tcl Procedures

You can use the predefined Tcl procedures in OMB*Plus:

- **OMBToSettableString:** Use this procedure when setting string values that contain single quotes that need to be escaped. The input for this procedure is a Tcl string and the output is a Tcl string with all single-quotes escaped.
- **OMBToTypeObjListString:** This procedure converts an input two-dimensional list to a comma-delimited string. For example, the procedure converts input in the form of

  ```
  {{<object_type> <name>} ... }
  ```
  to

  ```
  "<object_type> <name>,...".
  ```

- **OMBPageBreak:** This procedure displays the input string as a sequence of pages, with a pause after each page. When the output of a command is more than the page height, it may be difficult for screen reading software (used for accessibility) to read the whole text. This procedure may be used to break the output of a command into pages.

  The two inputs to the `OMBPageBreak` command are the number of lines to be displayed in a page and the string that is to be split into pages. The string may be the output of an OMB*Plus command. For example, the following command displays the output of the `OMBHELP OMBCREATE` command with 10 lines in a page.

  ```
  OMBPageBreak 10 [OMBHELP OMBCREATE]
  ```

  To display the next 10 lines of the output, press <Enter> on your keyboard.

  The `OMPageBreak` procedure is available for every OMB*Plus session.

3.1.3 Running Scripts in OMB*Plus

You can write scripts and run them in OMB*Plus. For examples of scripts you can write, see Chapter 4, "Sample Scripts".

Inside the interactive shell, type `source test.tcl` where `test` is the name of the script you want to run.

At the command line, type `OMBPlus.sh test.tcl` for scripts on UNIX and `OMBPlus.bat test.tcl` for scripts on Windows operating systems.

3.1.3.1 Locating Errors in Scripts and Multi-line Commands

OMB*Plus reports only the first error it encounters while executing a command. As soon as it encounters the first error, it stops processing the command and exits after reporting the error.

When an error occurs during the execution a multi-line OMB*Plus command, the error message that is displayed does not specify the exact line at which the error occurred. To determine the line at which the error occurred, use the following command immediately after you encounter an error:

```
OMB+> puts $errorInfo
```

3.2 OMB*Plus Commands

The sections that follow describe the types of commands that comprise the OMB scripting language.

- **Metadata Manipulation Language (MML) Commands:** Includes commands for creating, altering, deleting, and retrieving metadata objects.
- **Shell Commands:** Includes help and environment support such as `OMBDCC` and `OMBHELP`. Although these commands enable you to control the scripting environment, you cannot use them to edit the metadata.
- **Administrative Commands:** Fits the MML to the Warehouse Builder back end. For example, the commands `OMBCONNECT`, `OMBDISCONNECT`, `OMBCOMMIT`, or `OMBROLLBACK`.
- **Navigation Commands:** Enable you to navigate the Warehouse Builder repository just as you would navigate a UNIX file system.
- **Service Commands:** Enable you to start Warehouse Builder metadata services such as validation, compilation, deployment, and import or export.

3.2.1 Metadata Manipulation Language (MML) Commands

OMB*Plus enables you to create, modify, delete, and retrieve object metadata in Warehouse Builder design and runtime repositories. OMB*Plus commands work within the context of a first class object. For a list of first class objects, see "Warehouse Builder Metadata Objects" on page 3-8.

Table 3–2 lists the standard command names for MML.

Table 3–2 Metadata Manipulation Language Commands

Metadata Manipulation Language (MML)	Description
`OMBCREATE`	Creates a first class object.
`OMBDROP`	Deletes a first class object.
`OMBALTER`	Modifies a first class object.
`OMBRETRIEVE`	Retrieves information from a first class object.

The `OMBCREATE`, `OMBDROP`, `OMBALTER`, and `OMBRETRIEVE` commands accept only the object name as the main argument. Names identified by absolute or relative path are not accepted. To use these commands you must in the parent context of the object to be created, dropped, altered, or retrieved.

OMB*Plus executes commands like OMBCREATE, OMBALTER, and OMBDROP within a nested transaction.

OMB*Plus interprets clauses within a single command one by one, as illustrated by the following example:

```
OMBCREATE TABLE 'T1' \
   MODIFY COLUMN 'C1' RENAME TO 'C1_NEW' \
   ADD UNIQUE_KEY 'UK1' \
      SET REF COLUMNS ('C1_NEW', 'C2')
```

In the preceding example, OMB*Plus renames column C1 to C1_NEW when parsing the modify_column clause. In the last line, use the new name for the column, C1_NEW, to specify the referenced columns for the new unique key. For more details about synchronization of cached data, see "Synchronizing Cached Data with Repository Objects"

The OMBCREATE and OMBRETRIEVE commands synchronize only the first content object that they are currently working on. The OMBCREATE command synchronizes only the parent folder.

3.2.2 Examples

The following example lists the high-level scripting command syntax definitions for the OMBCREATE command:

```
OMBCREATE <fco_type> <fco_name> ( [ rename_clause ] [ properties_clause ] [ [ sco_
add_clause_for_alter ] | [ sco_modify_clause ] | [ sco_delete_clause ] ]* )1
rename_clause ::= RENAME TO <new_name>
sco_add_clause_for_alter ::= ADD <sco_type> <sco_name> [ OF parent_sco_clause ] [
AT POSITION <position> ] [ properties_clause ] [ references_clause ]*
sco_modify_clause ::= MODIFY <sco_type> <sco_name> [ OF parent_sco_clause ] ( [
rename_clause ] [ move_to_clause ] [ properties_clause ] [ references_clause ]* )1
move_to_clause ::= MOVE TO POSITION <position>
sco_delete_clause ::= DELETE <sco_type> <sco_name> [ OF parent_sco_clause ]
```

In the preceding example, the number 1 following a group of clauses enclosed by ( ) brackets indicates that you must specify at least one of the clauses.

You can specify a particular Warehouse Builder object by tracing the aggregation relationship from its parent first class object. You can also capture the association relationships by the references clauses. For example, getSCOClause, where sco_type is the second class object type.

Each action, create, alter, drop, or retrieve works only on the properties and the immediate children of the currently specified object. For example, the retrieve command on a table only enables you to access the properties of the table and the lists of column and constraint names owned by that table. To drill down to the detailed descriptions of the columns and constraints, you can call retrieve on these objects respectively.

The following statement retrieves the data type and length for a column in a view:

```
OMBRETRIEVE VIEW 'V1' COLUMN 'COL1' \
GET PROPERTIES (DATATYPE, LENGTH)
```

When you set and retrieve properties using the set_properties_clause and the get_properties_clause, you can type the property names in any order.

Physical names are used as object identifiers in scripting. Business names represent an object property. Business names are not used to identify objects. You can identify a cross-component first class object by a path notation.

```
/<project_name>/<module_name>/<fco_name>
or

../<module_name> <fco_name>
```

String values, including object names and string property values, must be enclosed in single quotes.

3.2.2.1 Warehouse Builder Metadata Objects

Use OMB*Plus to access and manipulate the following Warehouse Builder objects, also known as first class objects:

Activity Templates
Activity Template Folders
Advanced Queues
Alternative Sort Orders
Business Areas
Business Definition Modules
OBIEE Business Definition Modules
Business Presentation Modules
Calendars
Calendar Modules
CMI Definitions
CMI Modules
Collections
Configurations
Connectors
Control Centers
Cubes
Data Auditors
Data Profiles
Data Rules
Data Rule Modules
Deployments
Deployment Action Plans
Dimensions
Drill Paths
Drills To Detail
Experts
Expert Modules
External Tables
Flat Files
Flat File Modules
Folders
Functions
Gateway Modules
Iconsets
Item Folders
CTs
CT Folders
Lists Of Values
Locations

Mappings
Materialized Views
Nested Tables
Object Types
Oracle Modules
Packages
PLSQL Record Types
PLSQL Ref Cursor Types
PLSQL Table Types
Pluggable Mappings
Pluggable Mapping Libraries
Presentation Templates
Procedures
Process Flows
Process Flow Modules
Process Flow Packages
Projects
Queue Propagations
Queue Tables
Real Time Mappings
Registered Functions
Roles
SAP Modules
Sequences
Snapshots
Streams Capture Processes
Streams Queues
Tables
Table Functions
Time Dimensions
Transformation Modules
Transportable Modules
Users
Varying Arrays

For Oracle Modules, you can access only those Oracle modules designated as warehouse modules. You cannot access Oracle source modules using OMB*Plus.

3.2.3 Shell Commands

Shell commands provide you with an interactive interface to run all Warehouse Builder scripts and standard Tcl commands. OMB*Plus shell commands include: `OMBHELP`, `OMBCC`, `OMBDCC`, and `OMBENV`.

3.2.3.1 OMBHELP

Use the `OMBHELP` command to display help on Warehouse Builder commands. The help describes the purpose of the command, the syntax in BNF format, and each of the keywords or options. For complex commands, such as `OMBCREATE`, `OMBALTER`, and `OMBRETRIEVE`, you can specify an optional `fco_type` parameter. `OMBHELP` then displays the detailed syntax for that particular parameter type. Each command also provides specific options that enable you to display sub-sections of the help page.

The syntax for `OMBHELP` is:

```
help ::= OMBHELP <command_name> [ <command_specific_options> ] [DETAIL]
```

For example, OMBHELP OMBCONNECT displays the following:

```
OMBCONNECT
Purpose
To connect to OWB repository.
Syntax
OMBCONNECT <user>/<password>@<host:port:SID>
 where
   <user> is the OWB repository user name
   <password> is the OWB repository user password
   <host> is the name or IP address of the OWB repository host machine
   <port> is the numeric port for OWB repository database listener
   <SID> is the unique database identifier for OWB repository database
Notes:
   The connection to OWB repository will be established in single user mode.
```

If you type OMBHELP <command_ name> followed by [DETAIL], OMB*Plus displays the command purpose, prerequisites, syntax, descriptions for each keyword and parameter, and examples of how to use the command.

The OMBHELP command synchronizes only the only the FCO that you are currently working on.

3.2.3.2 OMBENV

The syntax for OMBENV is:

```
environment ::= OMBENV
```

This command lists the values for all Warehouse Builder-specific environment variables. Table 3–3 lists the environmental variables. To set an environmental variable, use the Tcl set command. Use unset to unset an environmental variable.

Table 3–3 Warehouse Builder Environment Variables

Environment Variable	Meaning	Possible Values
OMBTIMER	Enables timing on each Warehouse Builder scripting command. The time is logged to a log file and to the console or shell.	A Tcl boolean value.
OMBLOG	Stores the filename for Warehouse Builder log file.	A valid filename including its path.
OMBPROMPT	Indicates whether OMB*Plus will update the command prompt each time you call OMBCC.	A Tcl boolean value.
OMBCONTINUE_ON_ERROR	Ignores errors that occur in any command that is part of a script and moves to the next command in the script.	A Tcl boolean value.

3.2.4 Administrative Commands

Use these commands to perform administrative jobs on a Warehouse Builder repository. The following commands are available: OMBCONNECT, OMBDISCONNECT, OMBCOMMIT, and OMBROLLBACK.

```
connect ::= OMBCONNECT <username>/<password>@<host>:<port>:<sid>
disconnect ::= OMBDISCONNECT
```

```
commit ::= OMBCOMMIT
rollback ::= OMBROLLBACK
```

> **Note:** If you are running OMB scripts from OMB*Plus command line, then ensure that you commit (OMBCOMMIT) any changes before exiting the OMB*Plus session. If you exit from the session without a commit, then all the modifications since the previous commit will be lost.

3.2.5 Navigation Commands

You can use the following commands to navigate the Warehouse Builder repository in the same way you navigate a UNIX file system.

3.2.5.1 OMBCC

This command enables users to change context Up and Down the Warehouse Builder navigation tree. For example, when you type **...** the current context changes to the parent context. However, if the current context is a modified project, an error message prompts you to commit or rollback your changes. For example, to switch from the current project to another, use the command:

```
OMBCC '/PROJECT_NAME'
```

For example, if the name of the project is MATERIALS, then the command is:

```
OMBCC '/MATERIALS'
```

3.2.5.2 OMBDCC

This command shows you the current context and the context type. The syntax for OMBDCC is:

```
display_current_context ::= OMBDCC
```

3.2.5.3 OMBLIST

The child first class objects for folders are listed under OMBLIST. Using this command on folders describes only the folder properties. Note also that the list command allows name matching by regular expression. If you do not include the regular expression, then OMBLIST displays all objects sorted alphabetically.

The generic syntax for OMBLIST in a folder context is:

```
list_folder ::= OMBLIST ( <child_type1_plural> | …| <child_typeN_plural> ) [ name_
in_regexp ]
```

The name_in_regexp parameter represents a name in regular expression.

For example, under the root context you have:

```
list_root ::= OMBLIST PROJECTS [ name_in_regexp ]
```

The OMBLIST command synchronizes all parent-child relations in the navigation tree.

3.2.6 Service Commands

Service commands perform services such as batch operations on Warehouse Builder metadata. Table 3–4 contains a list of service commands and their descriptions.

Table 3–4 Service Commands

Command	Description
OMBCOMPILE	Use this command to compile folders or first class objects such as tables, views, sequences, dimensions, and cubes.
OMBDEPLOY	This command provides deployment service.
OMBIMPORT	This command provides the metadata import service. You can only invoke the OMBIMPORT command from the root context. The four available modes are: CREATE_MODE (CREATE), REPLACE_MODE (REPLACE), UPDATE_MODE (UPDATE), and MERGE_MODE (INCREMENTALUPDATE). The default mode, if not specified in the command, is CREATE_MODE.
OMBVALIDATE	Use this command to validate folders or first class objects such as tables, views, sequences, dimensions, and cubes.
OMBSTART	Use this command to execute objects from the control center.
OMBEXPORT	Exports current metadata, metadata definitions, or snapshot metadata to a metadata loader file.
OMBIMPACT	Use this command to fetch the impact of a change for an object.
OMBLINEAGE	Use this command to fetch the data lineage.

3.2.6.1 Synchronizing Cached Data with Repository Objects

When you start an OMB*Plus session, data about the objects is fetched from the OWB repository and cached in the OMB*Plus session. The cached data is synchronized with the data from the repository at certain predefined intervals.

To synchronize the objects in the repository and the OMB*Plus environment at any point of time, change to the parent context using OMBCC, and then use the command OMBLIST to refresh the objects within that context. For example, to refresh the tables within the parent context, use the command `OMBLIST TABLES`.

The extent to which the objects are synchronized depends on the OMB*Plus command that you execute. For example, some commands synchronize all the parent-child relationships in the navigation tree, whereas some commands synchronize only the first class object that they are currently working on.

- The `OMBLIST` command synchronizes all the parent-child relationships in the navigation tree.
- The `OMBCREATE` and `OMBRETRIEVE` commands synchronize only the first class object that they are currently working on.
- The `OMBCREATE` command synchronizes only the parent folder.

Consider the following example on synchronization of cached data. You open anOMB*Plus session and a Design Center session. In the Design Center, you delete a flat file module called FFM1. You then undelete FFM1 and commit the changes. In the OMB*Plus session, you perform the sequence of operations listed. The details of the result of the operation and the logic behind the result is explained:

1. `OMBCC FFM1`

   The context is changed to the module FFM1.

2. `OMBCREATE FLATFILE`

The flat file is not created because OMBCREATE synchronizes the parent folder. When the synchronization is performed, the parent folder is not found in the cache.

3. OMBLIST FLAT_FILE_MODULE

   FFM1 is listed as one of the modules. This is because OMBLIST synchronizes all the parent-child relationships in the navigation tree.

4. OMBCREATE FLATFILE

   The flat file is created. This is because the undelete is now reflected in the cache because of the OMBLIST command.

3.2.6.2 Changing the User Mode

When you connect to a repository, Warehouse Builder by default connects in the multiple user mode. To switch to single user mode, use the command OMBSWITCHMODE SINGLE_USER_MODE. If however, you are connected to the control center (OMBCONNECT CONTROL_CENTER), then you cannot switch to the single user mode.

If you have long running jobs, for example from process flows, then you might get an error message while trying to connect in single user mode. This is likely to occur when process flows are waiting for user intervention, or when the control center is restarted while process flows are running.

When you are unable to log into the single user mode, run the following script to obtain a list of pending deployment and execution requests for a particular workspace:

```
owb/rtp/sql/list_requests <WORKSPACE_NAME>
```

Based on this list, you can decide to either expedite the execution requests or stop those execution requests that need not be run currently.

To expedite execution requests, run the script:

```
owb/rtp/sql/expedite_exec_request <AUDIT_ID> <WORKSPACE_NAME>
```

To stop execution requests, run the script:

```
owb/rtp/sql/abort_exec_request <AUDIT_ID> <RETURN_NO> <RETURN_CODE> <WORKSPACE_
NAME>
```

Once you expedite or stop the pending execution requests, you will be able to log into the single user mode.

4

Sample Scripts

This chapter contains sample scripts that are appropriate in the context of a single OMB*Plus command statement. These examples provide the series of steps for using particular Warehouse Builder functionality.

This chapter includes sample scripts for the following tasks:

- Updating a Design Repository
- Reporting on Repository Objects
- Finding Invalid Objects
- Using OMB*Plus to Navigate Repositories

4.1 Updating a Design Repository

One possible use case is to perform mass update on repository metadata. Users can write the following script to add a primary key with local column ID for each table with name beginning in EDW inside the module MY_MODULE:

```
OMBCC MY_MODULE;
foreach tableName [OMBLIST TABLE EDW*] { \
OMBCREATE TABLE '$tableName' \
ADD PRIMARY_KEY '$tableName_PK' SET REFERENCE COLUMNS ('ID');}
```

We can build even more powerful and useful script using if-then-else:

```
foreach tableName [OMBLIST TABLE EDW*] { \
set columnList [OMBRETRIEVE TABLE '$tableName' GET COLUMNS]; # Use lsearch to
search for a name in a list
if {[lsearch $columnList 'ID'] == -1} {
      OMBCREATE TABLE '$tableName' \
         ADD COLUMN 'ID' \
            SET PROPERTIES (DATATYPE, LENGTH, NOT_NULL) VALUES \
               ('NUMBER', 10, 'true');
   }
}
```

The preceding script checks the list of tables which name begins with EDW whether each of them contains an ID column. If not, it will create an ID column for that table. Hence, executing the preceding script will guarantee that all tables with names beginning in EDW will have the ID column defined.

4.2 Reporting on Repository Objects

Another common use is for reporting purpose. The following script displays the properties of the table T1 and its column definitions on standard output:

```
#Displaying metadata of a table
puts -nonewline "Please enter the table name: " gets stdin tableName
puts ""
puts "Report on $tableName"
puts "====================================="
puts "Physical name = $tableName"
puts "Logical name = [lindex [OMBRETRIEVE TABLE '$tableName' GET \
PROPERTIES(BUSINESS_NAME)] 0]"
puts "Description = [lindex [OMBRETRIEVE TABLE '$tableName' GET \
PROPERTIES(DESCRIPTION)] 0]"
puts "-------------------------------------"
set columnList [OMBRETRIEVE TABLE '$tableName' GET COLUMNS]
set i 1
foreach colName $columnList {
set dt [lindex [OMBRETRIEVE TABLE '$tableName' COLUMN '$colName' GET \
PROPERTIES(DATATYPE)] 0]
   if { $dt == "VARCHAR2" } {
      set prop [OMBRETRIEVE TABLE '$tableName' COLUMN '$colName' GET
PROPERTIES(LENGTH, NOT_NULL)]
      puts "Column $i: $colName datatype=VARCHAR2 length=[lindex $prop 0] \
           not_null=[lindex $prop 1]"
   } elseif { $dt == "NUMBER" } {
      set prop [OMBRETRIEVE TABLE '$tableName' COLUMN '$colName' \
                GET PROPERTIES(PRECISION, SCALE, NOT_NULL)]
      puts "Column $i: $colName datatype=NUMBER precision=[lindex $prop 0] \
          scale=[lindex $prop 1] not_null=[lindex $prop 2]"
   } elseif { $dt == "DATE" } {
      set prop [OMBRETRIEVE TABLE '$tableName' COLUMN '$colName' GET \
PROPERTIES(NOT_NULL)]
      puts "Column $i: $colName datatype=DATE not_null=[lindex $prop 0]"
   } # end else
   incr i
}
```

A sample output is like the following:

```
Physical name = T1
Logical name = Table 1
Description = This is my first table.
====================================
Column: ID datatype=NUMBER precision=0 scale=0 not_null=1
Column: NAME datatype=VARCHAR2 length=30 not_null=1
Column: VALUE datatype=VARCHAR2 length=100 not_null=0
```

4.3 Finding Invalid Objects

Users can also take advantage of the validation service provided by scripting, like this:

```
set tableList [OMBLIST TABLES];
foreach tableName $tableList {
   if { [OMBCOMPILE TABLE '$tableName]' == "Invalid." } {
      set context [OMBDCC];
      puts "Table $context/$tableName is invalid.";
   }
}
```

The preceding script will tell users which table is invalid under the current module.

4.4 Using OMB*Plus to Navigate Repositories

Another scenario we present is for a disabled user that relies on OMB*Plus interactive shell (and also some screen reading software for the disabled) to navigate through a Warehouse Builder repository:

```
OMB+> OMBCONNECT owb/owb@localhost:1521:dev901
Connected.
OMB+> OMBLIST PROJECTS
DIM_TEST_PROJ MY_PROJECT PROJ_ENDTOEND PROJ_RELATIONAL TEST_DIM_PROJ
OMB+> OMBLIST PROJECTS .*RELATION.*
PROJ_RELATIONAL
OMB+> OMBCC 'PROJ_RELATIONAL'
Context changed.
OMB+> OMBDCC
PROJECT /PROJ_RELATIONAL
OMB+> set OMBPROMPT ON
ON
OMB+> OMBDCC
PROJECT /PROJ_RELATIONAL
/PROJ_RELATIONAL>
/PROJ_RELATIONAL> OMBLIST ORACLE_MODULES
WH
/PROJ_RELATIONAL> OMBCC 'WH'
Context changed.
/PROJ_RELATIONAL/WH> OMBLIST TABLES
PRODUCT PO
/PROJ_RELATIONAL/WH> OMBRETRIEVE TABLE 'PO' GET COLUMNS
OID PROD_ID ORDER_DATE CUSTNAME
/PROJ_RELATIONAL/WH> OMBCC '..'
Context changed.
/PROJ_RELATIONAL> OMBCC '..'
Context changed.
/>
/> OMBDISCONNECT
Disconnected.
```

Part III

Using Experts

This part consists of the following chapters:

5 About Experts

If you are an advanced Warehouse Builder user, you can design solutions that simplify routine tasks and implement best practices. You can develop these solutions, called experts, in the Expert Editor.

5.1 What Are Experts?

Experts are solutions that advanced users develop to automate routine or complex tasks using best practices.

For example, a common activity is extracting data from a flat file and loading that data into a table in Oracle Database. To accomplish this activity, users might take the following steps, in which they navigate a variety of user interfaces in Warehouse Builder:

1. Define a flat file module.
2. Identify the source file.
3. Specify the data format.
4. Define an external table.
5. Define an Oracle database module and location.
6. Define a mapping.
7. Validate, generate, and deploy all objects.
8. Execute the mapping.

To help users with this activity, you could design an expert that calls all the necessary user interfaces, provides customized instructions, and prompts users for input. In an expert, the steps are defined by **tasks** and the order of execution is defined by **transitions**.

Experts are reusable, shareable, and can access all areas of Warehouse Builder including user interfaces and the OMB*Plus scripting language. Experts can also call Java programs.

5.1.1 User's View of an Expert

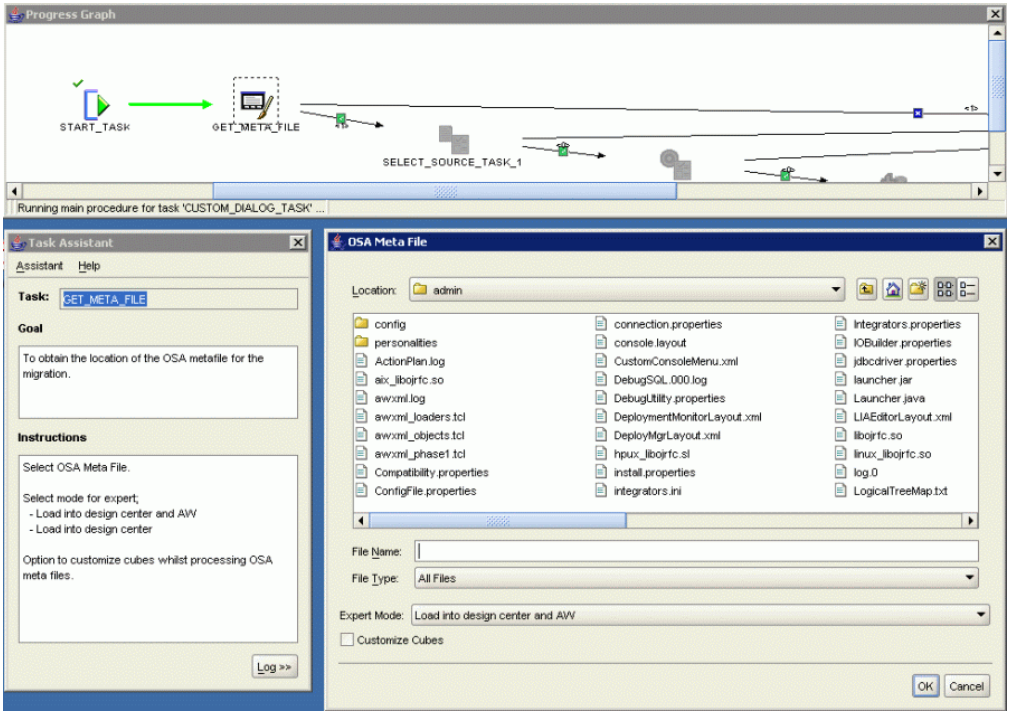
Figure 5–1 shows how an expert might look to a user. Three separate windows are displayed:

- **Progress Graph**: Displays the expert in the same format as it appears on the editing canvas, but the executed transitions and the current task are highlighted. This window serves the same purpose as bread-crumbs, tracking the user's

progress through the expert. It can also be a helpful debugging tool. You can set a property on the expert that controls whether the progress graph is displayed or hidden.

- **Task Assistant**: Displays the name of the current task, its goal, and instructions for using it. You provide this information on the property sheets for the task as part of its definition. You can set a property on the expert that controls whether the task assistant is displayed or hidden.
- **Task Display**: Tasks that obtain information from the user display various types of graphical user interfaces. Among these tasks are those that display Warehouse Builder components, such as the Object Editor, and those that provide basic functions, like the file selector shown in the figure. In this example, the user selects a file, and the filename is passed as an input parameter to the next task. You can also store the value in a variable to use later.

Figure 5–1 Execution of an Expert

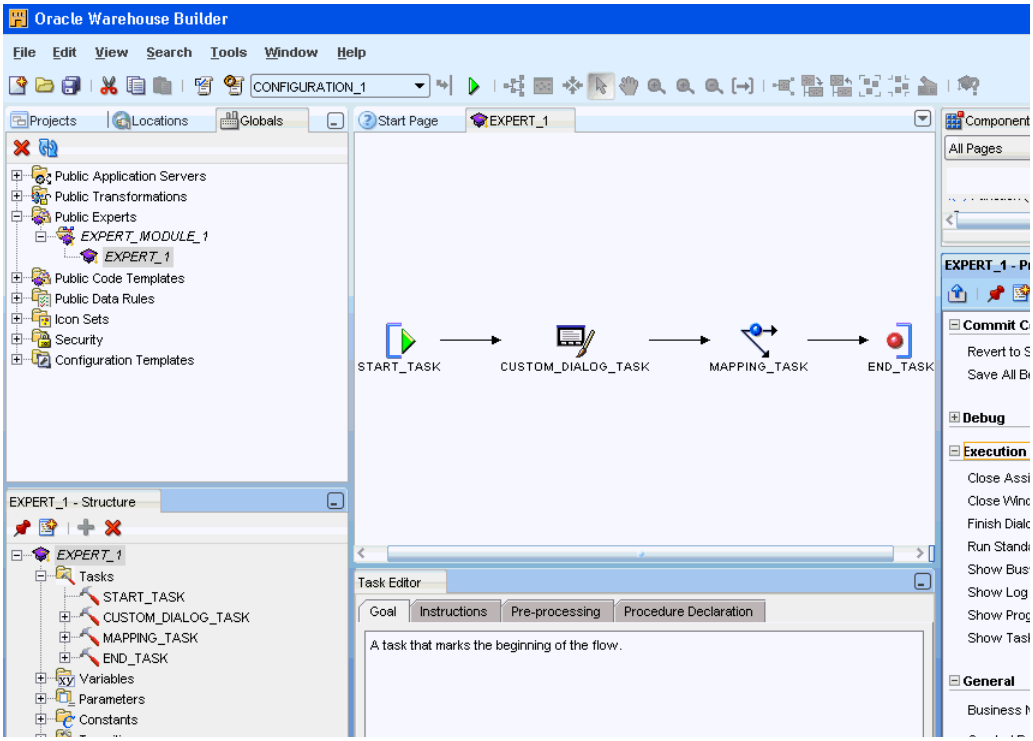


5.1.2 Developer's View of an Expert

The Expert Editor provides the canvas and the palette that you need to create, modify, and deploy experts.

Figure 5–2 shows the Expert Editor with a sample expert. Tasks appear as icons on the canvas. The arrows connecting the tasks are transitions. Transitions identify the order in which the tasks are executed.

Figure 5–2 Expert Editor



5.1.3 How Are Experts Different From Process Flows?

The Expert Editor is very similar to the Process Flow Editor. If you have created process flows, you will be able to adapt your knowledge very quickly to developing experts. However, there are important differences as well as important similarities.

- **Unit of Work**: In a process flow, the unit of work is an activity. In an expert, the basic unit of work is a task.
- **Transitions**: Both process flows and experts use transitions to connect the basic units of work. Transitions can be conditional.
- **Variables**: Both process flows and experts enable you to define local variables to pass values from one task to another.
- **End tasks**: Process flows have success, warning, and failure end activities, but experts have a single End task.
- **Subprocesses**: You can design a process flow to open other process flows, and you can design an expert to open other experts. In this use, they are called nested experts.
- **Code**: A process flow generates XML that conforms to the XPDL workflow standard. An expert generates Tcl.

6

Creating Experts

To create experts, you should have a working knowledge of Warehouse Builder, the Warehouse Builder scripting language, and Tool Command Language (Tcl), which is an open-source programming language.

> **See Also:** The Tcl Developer Xchange web site at `http://www.tcl.tk` for information about Tcl.

This chapter contains the following topics:

6.1 How to Create Experts

Before you begin the process of developing an expert, you should compose a plan that answers the following questions

- What job do you want the expert to accomplish?

  For example, refresh the data in a schema.

- What are the steps that you would take in Warehouse Builder to accomplish this job?

  For example, identify the locations of the source data files and the target schema, execute a mapping or process flow, and so forth.

- How flexible do you want the expert to be? That is, where can users provide input into the expert?

Experts can run without any user input at all, or they can prompt the user for input at every step. For example, an expert can refresh either a particular schema or the user's choice of a schema.

To define an expert, complete the following tasks:

1. Creating an Expert Object on page 6-2
2. Adding Tasks to the Expert Canvas on page 6-2
3. Adding Transitions to the Expert Canvas on page 6-4
4. Passing Data Values Among Tasks on page 6-4
5. Validating, Generating, and Starting Experts on page 6-6
6. Creating a Development Environment on page 6-6
7. Publishing Experts on page 6-7

6.2 Creating an Expert Object

Experts are organized in modules within a project.

Note: You can develop public experts directly from the Globals Navigator. Simply follow the steps below in the Globals Navigator instead of in the Projects Navigator. Administrators can also publish experts, as described in "Publishing Experts" on page 6-7.

To create an expert:

1. In the Projects Navigator, right-click **Experts** and select **New Expert Module**.

   The Create Expert Module dialog box is displayed.

2. Enter a name and description for the module.
3. Select the **Proceed to Wizard** box.

   The Create Expert dialog box is displayed.

4. Enter a name and description for the module.

   The Expert Editor is displayed. You can begin developing the expert.

To edit an expert module:

1. In the Projects Navigator, expand the Experts folder, then expand the module for the expert.
2. Double-click the expert.

   The Expert Editor is displayed.

6.3 Adding Tasks to the Expert Canvas

Tasks represent units of work for the expert. When you design an expert in Warehouse Builder, you select tasks from the component palette and drag them onto the canvas. The Start and the End tasks are already available on the canvas.

To add a task to an Expert:

1. Select a task from the palette and drag it onto the canvas.

Or

From the Expert menu, choose **Add**, then choose a category from the popup menu. Select a task from the category.

The editor displays the task on the canvas.

2. In the **Task Editor**, complete the tabs for the various properties of that task.

   These properties vary according to the type of task. All tasks have the Goal, Instructions, Pre-processing, and Post-Processing properties. Some tasks such as the Custom Dialog and OMB tasks include additional properties.

3. Use the Explorer and the Object Inspector to enter values for the parameters of the task.

   Tasks have input parameters and output parameters. Refer to "Passing Data Values Among Tasks" on page 6-4 for methods of providing values to the input parameters.

4. To change the name of the task or add a description, right-click the icon on the palette and select **Edit Details** from the popup menu.

   The Edit Task dialog box is displayed. Click **Help** for more information.

6.4 Adding Nested Experts to the Expert Canvas

You can create experts that perform relatively simple tasks, then use them as building blocks in developing more complex experts. When you add an expert to the canvas, it is called a **nested expert**. The nested expert functions as a single task, with one incoming transition and one or more outgoing transitions.

The existing flow becomes the **parent graph**, and the nested expert becomes a **child graph** on the canvas.

To add a nested expert:

1. From the Expert menu, select **Add**, then **Nested Experts**.

   Or

   Drag-and-drop experts from the Available Objects tab of the Explorer to the canvas.

   The Add Nested Experts dialog box is displayed.

2. Expand the module folders, then select one or more experts from the list. Click **OK**.

   The expert appears as a single icon on the canvas.

3. Draw transitions from a task to the expert, and from the expert to a task.

To view a nested expert:

The graphic toolbar provides icons for viewing the contents of a nested expert.

- To enlarge the expert icon so that you can see the individual tasks, click Expand Child Graph on the toolbar. This view of the nested expert is similar to the Bird's Eye View of the canvas.

  To shrink the nested expert back to an icon, click the arrow on the right side of its title bar.

- To fully examine the tasks and transitions in the nested expert, click Visit Child Graph on the toolbar. The canvas displays only the nested expert, and its

properties are shown in the Explorer, Object Details, and Task Editor windows. You have read-only access to a nested expert.

To shrink the nested icon back to a component in the larger expert, click Return to Parent Graph on the toolbar.

To edit a nested expert:

Open the nested expert in its own Expert Editor and make your changes. These changes are reflected in the parent expert as soon as you save the changes, with one exception: Changes to the expert parameters.

If you make changes to the parameters of the nested expert, you must delete and add the nested expert in the parent.

6.5 Adding Transitions to the Expert Canvas

Transitions indicate the sequence for executing the tasks in an expert. You can use conditional transitions to execute a task based on the completion state of the preceding task. By using conditional transitions, you can develop an expert that handles errors gracefully and provides alternative actions when users cancel a task.

A single task can have multiple outgoing transitions, but only one will be executed. If more than one transition evaluates to true, then the first one alone is executed.

To connect two tasks with a transition:

1. Add one or more tasks to the canvas, following the steps in "Adding Tasks to the Expert Canvas" on page 6-2.
2. Click the Select tool on the toolbar.
3. Position the pointer over the previous task so that it has an arrow shape. Click and drag the pointer to the next task.

   Notice that a plus sign appears under the pointer, then changes to a circle as the pointer approaches a next task.

   The editor displays an arrow between the two tasks, assigns a default name to the transition, and displays the transition in the Explorer and in the Object Selector.
4. If you want execution of the next task to be conditional, then assign a condition to the transition in the Object Details window. Table 6–1 describes the conditions.
5. After you have connected all the tasks, click the Auto Layout tool to arrange the tasks in the order of execution and spaced evenly across the canvas.

Table 6–1 Types of Conditions for Transitions

Condition	Continuation to the Next Task
Success	Only if the preceding task ends in success.
Error	Only if the preceding task ends in error.
Cancel	Only if the user cancels the preceding task.
Complex Condition	Only if the conditions you specified are true.

6.6 Passing Data Values Among Tasks

Tasks have both input parameters and output parameters.

- Input parameters affect the execution of the task. You can provide these values, or you can design the expert so that the user provides them.
- Output parameters are returned by the task. These values are the result of the work performed by the task.

You can pass data values among tasks either by binding the parameters or by using constants and variables.

6.6.1 Binding Input Parameters

You can bind the input parameters of a task to the output parameter of another task, to a global constant, or to a global variable. Binding is the easiest method of passing values to input parameters.

To bind the parameters:

1. Create the source and the target tasks.
2. In the Expert Explorer, select an input parameter for the target task.
3. In the Object Details window, click the **Binding From** field and select an output parameter, a variable, or a constant from the popup menu.

6.6.2 Using Constants

You can use the following predefined constants in any task:

- `OMB_CURRENT_PROJECT`: Stores the name of the current project.
- `OMB_CURRENT_SESSION`: Stores the identifier of the current session.
- `OMB_CURRENT_USER`: Stores the user ID of the current user.

These constants appear in the Explorer under the Constants folder.

6.6.3 Using Variables

You can use custom variables to store the values of output parameters. This is particularly useful when using Custom Dialog tasks, because the `GUI_RETURN_VALUE` output parameter is an array that functions like a hash table. It must be parsed before the individual values can be passed to input parameters. You may also want to use variables when passing a single value to multiple tasks, or when passing a value further down the flow.

To create a variable:

1. In the Explorer, select the Variables folder.
2. Click the **Create** icon at the top of the Explorer.

   A variable named `VARIABLE` appears in the Variables folder.
3. Select `VARIABLE` and use the Object Details window to change its name, data type, and other parameters.

Or, you can use Tcl commands to declare variables in the Pre-processing or Post-processing tabs of the Task Editor.

To assign a value to a variable:

Use a Tcl variable assignment statement in the Pre-processing or Post-processing tabs of the Task Editor. The following are some examples:

This example assigns the value of the RETURN_VALUE output parameter to a variable named THIS_OBJECT_NAME.

```
set THIS_OBJECT_NAME $RETURN_VALUE;
```

The next example assigns the value of the first component (COMP1) of the GUI_RETURN_VALUE parameter to a variable named THIS_TABLE.

```
set THIS_TABLE $GUI_RETURN_VALUE(COMP1)
```

To use a variable:

You can bind an input parameter of a task to a variable. The variables appear in the popup menu in the Object Details window.

You can also use the variables in any Tcl commands on the Pre-processing and Post-processing tabs of any task, and on the Main tab of an OMB task.

6.7 Validating, Generating, and Starting Experts

You can validate, generate, and start an expert from the Expert Editor or from the Design Center.

To validate, generate, or start an expert from the Expert Editor:

From the Expert menu, choose the **Validate**, **Generate**, or **Start** command.

or

Click the Validate, Generate, or Start icon from the toolbar.

The Message tab of the Compilation Results window displays any validation errors and warnings. When you select a message, Warehouse Builder selects the errant task or transition on the canvas, the Explorer, and the Object Details window, so that you can correct the problem.

To validate, generate, or start an expert from the Design Center:

1. In the Projects Navigator, expand the Experts folder for the project, then expand the module folder.
2. Right-click the expert, then choose the **Validate**, **Generate**, or **Start** command from the popup menu.

   or

   Select the expert, then choose the **Validate**, **Generate**, or **Start** command from the Design menu.

6.8 Creating a Development Environment

Experts have several settings that affect their behavior while running. You will probably want to set them one way while you are developing an expert, and another way when you or other users are using the expert to accomplish real work.

To set the expert parameters for development:

1. In the Expert Explorer, select the expert at the very top of the tree.
2. In the Object Details window, select the following settings:
   - **Show Progress Graph**
   - **Show Task Assistant**

- **Show Log Window**
- **Save All Before Start**
- **Logging**

The two logging parameters (Show Log Window and Logging) display the scripting messages in the Task Assistant when you run the expert. These messages may help you diagnose any problems.

6.9 Publishing Experts

The settings that simplify testing and debugging experts are not appropriate when running the expert to accomplish work.

To set the expert parameters for use:

1. In the Expert Explorer, select the expert at the very top of the tree.
2. In the Object Details window, deselect the following settings:
   - **Show Log Window**
   - **Save All Before Start**
   - **Logging**
3. Select the following settings:
   - **Close Windows on Execution**
   - **Finish Dialog on Completion**
   - **Run Standalone**
   - **Close Assistant on Completion**
   - **Revert to Saved on Error**
4. Enter a name for the expert in the Menu Item Display String field.

To publish an expert:

If you have administrative privileges in Warehouse Builder, you can make experts available to other users in any of these places:

- **Globals Navigator**: Copy and paste an expert from the Projects Navigator to the Global Explorer into a module in the Public Experts folder.
- **Projects Navigator**: Right-click a folder and choose **Add/Remove Experts Here**.
- **Design Center Menu**: On the Tools menu, choose **Add/Remove Experts Here**.
- **Windows Program Menu**: Create a batch file that can be run from the Windows Program menu, as described in "Running an Expert From a Batch File" on page 6-8.

Experts that appear in the Public Experts folder are available to all users for all projects. Users can scan the contents of the Public Experts folder to find experts relevant to their objectives in Warehouse Builder.

Alternatively, you can make experts more accessible to end users by customizing the menus in the Design Center to include public experts. For example, if you designed an expert for creating a common type of mapping, you can customize the right-click menu for the mapping folder to include this specialized expert.

6.10 Running an Expert From a Batch File

You can create a batch file that enables users to run an expert without opening the Warehouse Builder Design Center. Take these steps, which are explained in more detail in the following paragraphs:

1. Create a Tcl script that starts the expert.
2. Create a batch file that opens OMB*Plus and passes it the name of the Tcl file.
3. For Windows platforms, create a shortcut to the batch file on the Start menu or on the desktop.

To create a Tcl script:

Use a text editor to create a script containing the following Warehouse Builder scripting commands:

- `OMBCONN`: Connects a user to a Warehouse Builder repository.
- `OMBCC`: Sets the context to the location of the expert.
- `OMUSTART EXPERT`: Starts the expert.
- `OMBDISCONNECT`: Closes the session.

Example 6–1 connects a user named `SCOTT` to the `GCCREP` repository on a host named `SCOTT-PC`. The expert is named `REFRESH_DATA_EXPERT`, and is located in `SALES_PROJECT` in `EXPERT_SALES_MODULE`.

For detailed information about these commands, refer to the *Oracle Warehouse Builder API and Scripting Reference*.

Example 6–1 Sample Tcl Script for Starting an Expert

```
OMBCONN scott/tiger@scott-pc:1521:orcl USE REPOS 'GCCREP'
OMBCC '/SALES_PROJECT/EXPERT_SALES_MODULE'
OMUSTART EXPERT 'REFRESH_DATA_EXPERT'
OMBDISCONNECT
```

To create a batch file:

Use a text editor to create a batch file that opens OMB*Plus and passes it the Tcl file. Be sure to specify the full path names, as shown in the following example:

```
c:\oracle\product\BiToolsHome_1\owb\bin\win32\OMBPlus.bat c:\owb_scripts\Update_
Sales_Data.tcl
```

To create a shortcut:

Search Windows Help for instructions for adding a program to the Start menu. Create a shortcut to the batch file.

7

Creating and Managing Custom Objects and Properties

This chapter describes how to extend the workspace by creating custom objects and custom properties. This chapter includes the following topics:

7.1 Extending the Workspace With User Defined Objects

You may encounter scenarios that require you to extend the OWB object model with new types of objects or properties. For example, as you use Warehouse Builder in conjunction with other applications, you may want to document how the data and metadata you manage in Warehouse Builder interacts with other applications.

Oracle Warehouse Builder supports user-defined objects (UDOs) and user-defined properties (UDPs) that allow you to represent these new objects or to annotate pre-defined Warehouse Builder objects with new properties.

You can specify icons to represent user-defined objects, edit user-defined objects and properties with a basic GUI editor in Design Center, and reference them in OMB*Plus scripts. Combining UDOs, UDPs with experts and scripting is one powerful way to extend Warehouse Builder's feature set.

You can also establish relationships between UDOs and other objects, and perform lineage and impact analysis that includes the UDOs.

For example, as you use Warehouse Builder in conjunction with other applications, you may want to document how the data and metadata you manage in Warehouse Builder interacts with other applications. To facilitate this documentation, you can introduce new metadata objects and associate those objects with existing workspace objects. These custom objects appear in the Design Center with the icon of your choice. You can edit them with a basic editor and perform lineage and impact analysis.

Users with administration privileges can extend the workspace by adding new properties and objects.

- **Adding New Properties to Workspace Objects**: Each workspace object has a pre-defined property set. If you would like to add new properties specific to your environment, this can be done with UDP's. (For example, you might like to add a

property called Design Notes for each of your metadata objects.) Once defined, UDP's are treated the same as pre-defined properties.

- **Adding UDOs to the Workspace**: You can introduce new types of objects to the workspace by defining UDOs which follow the general rules for object locking, multiuser access, transactions, security, and snapshots. You can also import and export UDOs and UDPs using the Metadata Loader (MDL). Note that after you define a new UDO, you can use it in scripting as well.

For the sake of clarity, this chapter refers to the objects native to the workspace as *workspace objects*. Any objects you introduce to the workspace are *UDOs* and any new properties are *UDPs*.

7.1.1 Using OMB*Plus Scripts to Specify UDOs and UDPs

OMB*Plus scripts enable you to define new objects, add and remove properties, as well as view the attributes for existing objects. The syntax is case sensitive and must be in upper case. While creating UDOs and UDPs, follow the guidelines in "Naming Conventions for UDOs and UDPs" on page 7-3.

OMBDEFINE

`OMBDEFINE CLASS_DEFINITION` enables you to create new object definitions in the workspace.

To create a new module definition, use the following command:

```
OMBDEFINE MODULE CLASS_DEFINITION 'UD_TABLEMODULE' SET PROPERTIES
   (BUSINESS_NAME, PLURAL_NAME) VALUES ('Table Module', 'Table Modules')
```

This creates a new module definition called `UD_TABLEMODULE`.

OMBREDEFINE

`OMBREDEFINE CLASS_DEFINITION` enables you to redefine a UDO.

To create a UDP on the Dimension object, use the following command:

```
OMBREDEFINE CLASS_DEFINITION 'DIMENSION'
     ADD PROPERTY_DEFINITION 'UD_DOCID' SET PROPERTIES (TYPE, DEFAULT_VALUE)
     VALUES ('INTEGER', '100')
```

This adds a new property definition called `UD_DOCID` to class definition `DIMENSION`.

The following command adds a new property definition for notes for the `COLUMN` type. Because columns exist in tables, views, materialized view, external tables and sequences, the following command adds the definition of this property to columns for all of those metadata objects:

```
OMBREDEFINE CLASS_DEFINITION 'COLUMN'
     ADD PROPERTY_DEFINITION 'UD_COL_NOTE' SET PROPERTIES (TYPE, DEFAULT_VALUE)
     VALUES ('STRING', 'notes')
```

When you create and save a new property definition, OMB*Plus performs the following validations:

- A user access check ensuring that you have single-user access to the current workspace.
- A name space check ensuring that you did not define two identically named property definitions within the same class hierarchy.

- A property value check ensuring that you defined default values consistent with the data types that you specified.

To change the name or the default value of a given property definition, use a command as follows:

```
OMBREDEFINE CLASS_DEFINITION 'TABLE' MODIFY PROPERTY_DEFINITION 'UD_TBL_NOTE'
     SET PROPERTIES (DEFAULT_VALUE, BUSINESS_NAME)
     VALUES ('99', 'Table Note')
```

To delete a UDP, use a command such as

```
OMBREDEFINE CLASS_DEFINITION 'TABLE' DELETE PROPERTY_DEFINITION 'UD_TBL_NOTE'
```

which deletes the UD_TBL_NOTE property definition from the Table class. Deleting a UDP is a destructive action and should generally be done with caution because it cannot be undone. It renders irretrievable all custom property values made for this property definition in your workspace.

OMBDESCRIBE

You can use `OMBDESCRIBE` on any class definition to view the attributes for a metadata element. Among other tasks, use `OMBDESCRIBE` to list all the property definitions including the UDPs for a given object type. For instance, the following command lists the UDPs for dimensions:

```
OMBDESCRIBE CLASS_DEFINITION 'DIMENSION' GET USER_DEFINED PROPERTY_DEFINITIONS
```

You can also use `OMBDESCRIBE` to inspect the properties of a property definition. For instance, for a UDP called `UD_DOCID`, you can discover its data type, default value, and business name using the following command:

```
OMBDESCRIBE CLASS_DEFINITION 'DIMENSION' PROPERTY_DEFINITION  'UD_DOCID'
     GET PROPERTIES (TYPE, DEFAULT_VALUE, BUSINESS_NAME)
```

7.1.2 Naming Conventions for UDOs and UDPs

It is mandatory to include the prefix UD_ while naming UDOs and UDPs. This ensures that the names of UDOs and UDPs are not identical to the names of predefined objects in the workspace.

> **Note:** If in a previous release you named a UDP with the prefix UDP_, then it is still valid. However, for all subsequent UDOs and UDPs, use the UD_ prefix.

7.2 Adding New Properties to Workspace Objects

To define new properties on Workspace objects, complete the following steps:

1. Carefully plan the new additions to the workspace.

   If possible, you should define all user-defined properties into the workspace before enabling end users to access it. In doing so, you avoid the task of supplying values for UDPs on existing objects.

2. Log in as the workspace owner, in single user mode.

   If another user is logged on within the GUI or OMB*Plus, then you will be prevented from running commands that alter the structure of the workspace.

If already logged into the Design Center, then you can start OMB*Plus from the main menu of Design Center. On the main menu, click **View**, and select **OMB*Plus**. To ensure that other users do not access the workspace, issue the command `OMBSWITCHMODE SINGLE_USER_MODE`.

3. Use the command OMBREDEFINE on the workspace object to which you want to add a custom property definition.OMBDEFINE FOLDER_DEFINITION

   For examples on how to use this command, see the section OMBREDEFINE on page 7-2.

4. To view the changes in OMB*Plus, use the command OMBDESCRIBE.
5. Use the command OMBCOMMIT to save the changes.
6. After you commit the changes, users can view and assign values to the new property in the graphical user interface.
7. Notify users that they can log in to the Design Center.

7.2.1 Creating UDPs: An Example

To create UDPs for an object, complete the following steps:

1. Log in to the client as an administrator.
2. Open the OMB*Plus view from the main menu of the Design Center. To do this, click **View**, and select **OMB*Plus**.
3. Ensure that you are in single user mode. You can verify this with the command `OMBDISPLAYCURRENTMODE`. If you are in multiple user mode, then switch to single user mode by using the command:OMBREDEFINE CLASS_DEFINITION 'VIEW

   ```
   OMBSWITCHMODE SINGLE_USER_MODE
   ```

4. In the OMB*Plus client, enter the following command to create four UDPs for the object `View`:

   ```
   OMBREDEFINE CLASS_DEFINITION 'VIEW' \
   ADD PROPERTY_DEFINITION 'UD_OWNER' SET PROPERTIES \
   (TYPE, DEFAULT_VALUE, BUSINESS_NAME) VALUES \
   ('STRING', 'REP_OWNER', 'Object Owner')

   OMBREDEFINE CLASS_DEFINITION 'VIEW' \
   ADD PROPERTY_DEFINITION 'UD_FILE' SET PROPERTIES \
   (TYPE, DEFAULT_VALUE) VALUES ('FILE', 'C:\\vw.sql')

   OMBREDEFINE CLASS_DEFINITION 'VIEW' \
   ADD PROPERTY_DEFINITION 'UD_LINK' SET PROPERTIES \
   (TYPE, DEFAULT_VALUE) VALUES ('URL', 'http://www.oracle.com')

   OMBREDEFINE CLASS_DEFINITION 'VIEW' \
   ADD PROPERTY_DEFINITION 'UD_VERSION' SET PROPERTIES \
   (TYPE, DEFAULT_VALUE) VALUES ('DATE', '2006-1-7')

   OMBSAVE
   ```

   This creates the following UDPs: `UD_OWNER`, `UD_FILE`, `UD_LINK`, and `UD_VERSION`.

   Note that the valid UDP types are: integer, string, float, double, date, timestamp, boolean, long, file, and url.

5. From the Projects Navigator create a view in any module.

6. Open the property inspector for the view. To do this, select the view and from the main menu, click **View**, and then select **Property Inspector**.
7. The user defined properties are listed on the property inspector.

   You can modify the values of any of the UDPs from the property inspector.

To remove a UDP from the workspace, use the `DELETE` clause. For example, to delete UD_VERSION, use the following command:

```
OMBREDEFINE CLASS_DEFINITION 'VIEW' DELETE PROPERTY_DEFINITION 'UD_VERSION'
```

7.3 Adding UDOs to the Workspace

UDOs are objects that you define and add to the workspace in addition to existing workspace objects.

All UDOs must belong to a module, and the module itself is a UDO. This module acts as the topmost container holding other objects within it. A module can contain folders, first class objects (FCOs), and second class objects (SCOs). Similarly, a folder can contain other folders, FCOs, and SCOs. An FCO can contain one or more SCOs.

UDOs exhibit a parent-child relationship. The module is the topmost parent. An FCO within a module is a child element of the module. Similarly, an SCO within an FCO is a child element of the FCO. For example, an Oracle module is a parent module. A table within this module is an FCO and a column within the table is an SCO.

To define new objects for the workspace, complete the following steps:

1. Carefully plan the new additions to the workspace.

   Before you begin, fully review the remainder of this chapter and become familiar with the necessary scripting commands.
2. Log in to the client as an administrator and in single user mode.
3. Design the UDO based on the steps described in "Writing Scripts to Define UDOs".
4. Use the OMBDEFINE command to create a new module definition, and FCOs and SCOs within that module. Use the OMBREDEFINE command to make any changes to the UDOs or to set properties for these objects.

   Once you create the UDO through scripting, you can use the graphical user interface to create and edit the objects it contains.
5. Log in to the Design Center and view the new objects as described in "Working with UDOs and UDPs" on page 7-11.

   Verify that the new objects display as intended.
6. (Optional) Assign a new icon to the UDO, as described in"Creating New Icons for Workspace Objects" on page 7-12.
7. Save the changes using the command OMBCOMMIT.
8. Notify users that they can log in to the client.

7.3.1 Writing Scripts to Define UDOs

To define a UDO, write a script that completes the following steps:

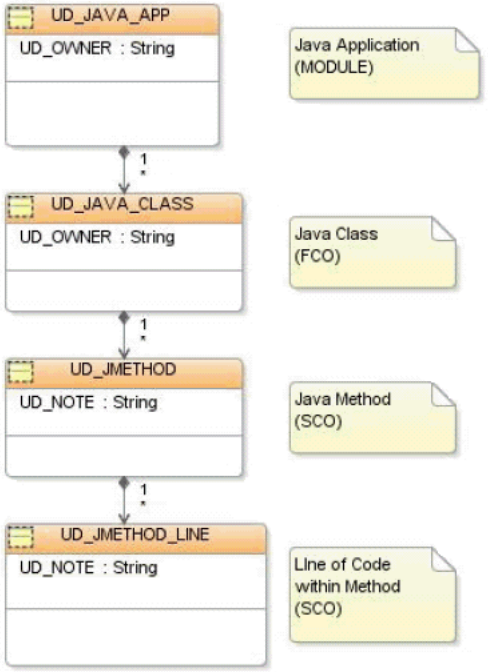
- **Create a user defined module**: This will be the parent module.
- **Define the object type**: Define the module to be a folder. This folder is an object that can hold other objects.

- **Define FCOs and SCOs**: Create user-defined FCOs and SCOs for the UDO, define the parent-child relationship between the FCOs and the SCOs, and assign physical names to these objects. For example, `UD_WORKBOOK` is a valid physical name. You can also indicate a business name and plural name, both of which are displayed in the Design Center and in editors. Continuing the previous example, `Workbook` and `Workbooks` are likely entries for the business name and plural name, respectively. If you do not specify these values, then they default to the physical name.
- **Define object properties**: Define the properties for all the objects you create. Some properties, such as Name, Business_Name, Plural_Name, and Description, are assigned automatically to any newly created object.
- **Add component definition**: All parent objects need to be assigned a component definition. The child elements have to be added to the component definition. The component definition determines the lifetime of child elements.For example, a column cannot exist if the parent table is deleted. Note that a component definition includes all the SCOs for the parent FCO, and all the SCOs owned by the first level SCOs and so on. If the child type is a FCO, then it must be added to the FOLDER_DEFINITION of the parent.
- **Define association types**: Create association types to indicate the types of relationships a UDO can have with workspace objects and other UDOs. You need to perform this step only if you want end users to later relate the UDO to specific instances of objects. For instance, in your script you could associate the UDO with tables and views. In the Design Center, end users could then relate instances of the UDO with specific tables and views. Warehouse Builder displays these relationships in impact and lineage analysis reports.
- **Assign icons (optional)**: See "Creating New Icons for Workspace Objects" on page 7-12.
- Save the changes.

7.3.2 Creating UDOs: An Example

This section provides an example to create UDOs modeled on a Java application. The Java application acts as a module. This module contains classes (FCOs), and those classes contain methods (SCOs). Within a method, you can model the lines of code. From a business standpoint, this is of interest because a particular line of code in an application may be impacted by a change in a database table if it is used within a SQL (JDBC) statement.

Figure 7–1 Structure of the UDOs



To create the UDOs, perform the following steps:

1. Log in to the Warehouse Builder client as an administrator and open the OMB*Plus window. Make sure that you are logged in single user mode.
2. First create a module definition and set properties for this module:

   ```
   OMBDEFINE MODULE CLASS_DEFINITION 'UD_JAVA_APP' \
   SET PROPERTIES (BUSINESS_NAME, PLURAL_NAME) \
   VALUES ('Java Application', 'Java Applications')
   ```

   This defines the module definition and sets certain properties common to all objects. BUSINESS_NAME is the user-friendly name for an object. If the Naming mode preference for the Design Center is switched to Business mode, then the value set for BUSINESS_NAME is displayed for the object. PLURAL_NAME is the label that is used to show where multiple instances of an object are shown, such as the label used for a tree node in the Design Center that contains several instances of the object.
3. Now create a folder definition with the same name as the module so that the module assumes the role of a folder:

   ```
   OMBDEFINE FOLDER_DEFINITION 'UD_JAVA_APP'
   ```

4. Now create an FCO:

   ```
   OMBDEFINE FIRST_CLASS_OBJECT CLASS_DEFINITION \
   'UD_JCLASS' SET PROPERTIES (BUSINESS_NAME, PLURAL_NAME) \
   VALUES ('Java Class File', 'Java Class Files')
   ```

5. Add the FCO as a child of the folder class:

   ```
   OMBREDEFINE CLASS_DEFINITION 'UD_JAVA_APP' \
   ADD CHILD_TYPE 'UD_JCLASS'
   ```

6. Create a component definition for the FCO:

```
OMBDEFINE COMPONENT_DEFINITION 'UD_JCLASS'
```

7. Add the component definition to the folder definition:

```
OMBREDEFINE FOLDER_DEFINITION 'UD_JAVA_APP' \
ADD 'UD_JCLASS'
```

8. Create an SCO and set its properties:

```
OMBDEFINE SECOND_CLASS_OBJECT \
CLASS_DEFINITION 'UD_JMETHOD' \
SET PROPERTIES (BUSINESS_NAME, PLURAL_NAME) \
VALUES ('Method', 'Methods')
```

9. Add the SCO as a child of the FCO:

```
OMBREDEFINE CLASS_DEFINITION 'UD_JCLASS' \
ADD CHILD_TYPE 'UD_JMETHOD'
```

10. Add the SCO to the component definition:

```
OMBREDEFINE COMPONENT_DEFINITION 'UD_JCLASS' \
ADD 'UD_JMETHOD'
```

11. Create an SCO and set its properties:

```
OMBDEFINE SECOND_CLASS_OBJECT \
CLASS_DEFINITION 'UD_JMETHOD_LINE' \
SET PROPERTIES (BUSINESS_NAME, PLURAL_NAME) \
VALUES ('Java Method Line', 'Java Method Lines')
```

12. Add this SCO as a child of the initially created SCO:

```
OMBREDEFINE CLASS_DEFINITION 'UD_JMETHOD' \
ADD CHILD_TYPE 'UD_JMETHOD_LINE'
```

13. Add this SCO to the component definition:

```
OMBREDEFINE COMPONENT_DEFINITION 'UD_JCLASS' \
ADD 'UD_JMETHOD_LINE'
```

This creates the following UDOs:

- A module folder called UD_JAVA_APP
- An FCO named UD_JCLASS, within the module
- An SCO named UD_JMETHOD, which is the child of UD_JCLASS
- Another SCO named UD_JMETHOD_LINE, which is the child of UD_JMETHOD

You can access the UDOs from the Projects Navigator under the User Defined Modules icon. To create a new instance of the UDO, right-click the UDO and select **New**. You can create new modules and FCOs as well as edit these modules and FCOs.

> **Note:** If you cannot see the newly created UDOs in the Projects Navigator, shut down Warehouse Builder completely, and then start it up again, saving any changes if prompted.

7.3.3 Associating UDOs with Objects

UDOs can be associated with other objects. By creating these associations, UDOs become a part of Lineage and Impact Analysis diagram just like any other object.

Associating a Java Application with a Table

This example associates the SCO, UD_JMETHOD, with one or more tables. This is modeling the fact that a method could be referencing tables in JDBC calls.

To associate the Java method to table, use the command:

```
OMBDEFINE ASSOCIATION_DEFINITION 'UD_XJMETHOD2TABLE' \
SET PROPERTIES (CLASS_1,CLASS_2,ROLE_1,ROLE_2 \
,ROLE_1_MAX_CARDINALITY,ROLE_1_NAVIGABLE) \
VALUES ('UD_JMETHOD','TABLE','TABLEUSED','JM2TABLE' \
,'INFINITE','true') ADD DEPENDENCY_DEFINITION 'DATAFLOW'
```

CLASS_1 and CLASS_2 can be any classes (FCO or SCO). At least one of the classes should be a user defined class. The other class can be either a user defined class or one of the main Warehouse Builder classes, such as table or column. In this example, the association is between the UDO UD_JMETHOD, and table.

Role_1 and Role_2 are the names you use to identify Class_1 from the point of view of this association. A class may have multiple associations and it plays a role in each one of them.

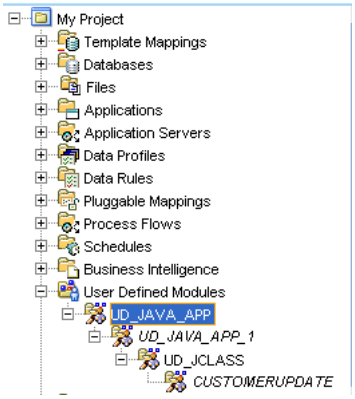
MAX_CARDINALITY allows you to limit the number of objects of that class which participate in that association. For example, consider the association between uniqueKey and foreignKey. The max_cardinality of uniqueKey is 1, because a given foreignKey object can be associated with at most one uniqueKey object. MAX_CARDINALITY can be set to any positive integer, or the reserved word INFINITE.

ROLE_1_NAVIGABLE is used by Lineage/Impact analyzer. If set to TRUE, it means that the analyzer can traverse the association in either direction between Class_1 and Class_2. If the property is set to FALSE, it means that the analyzer can traverse the relationship from Class_1 to Class_2, but not the other way around.

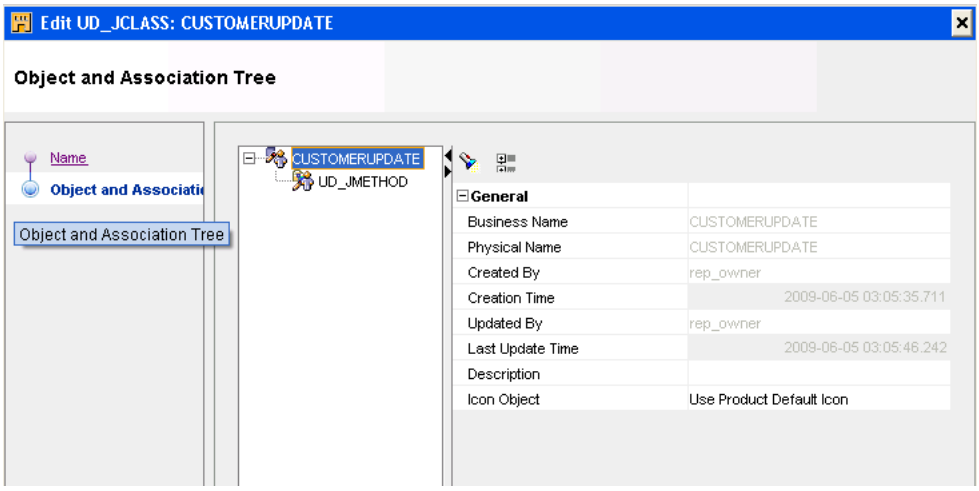
When you set the DEPENDENCY_DEFINITION parameter to DATAFLOW, the association participates in the dataflow impact and lineage analysis.

To associate the UDO to a table, complete the following steps:

1. In the Projects Navigator, expand the node User Defined Modules.
2. Right-click **UD_JAVA_APP** and select **New UD_JAVA_APP**.
3. Specify a name for the application.
4. Right-click **UD_JCLASS** and select **New UD_JCLASS**.
5. Specify a name for the Java class. In the example, it is CUSTOMERUPDATE.



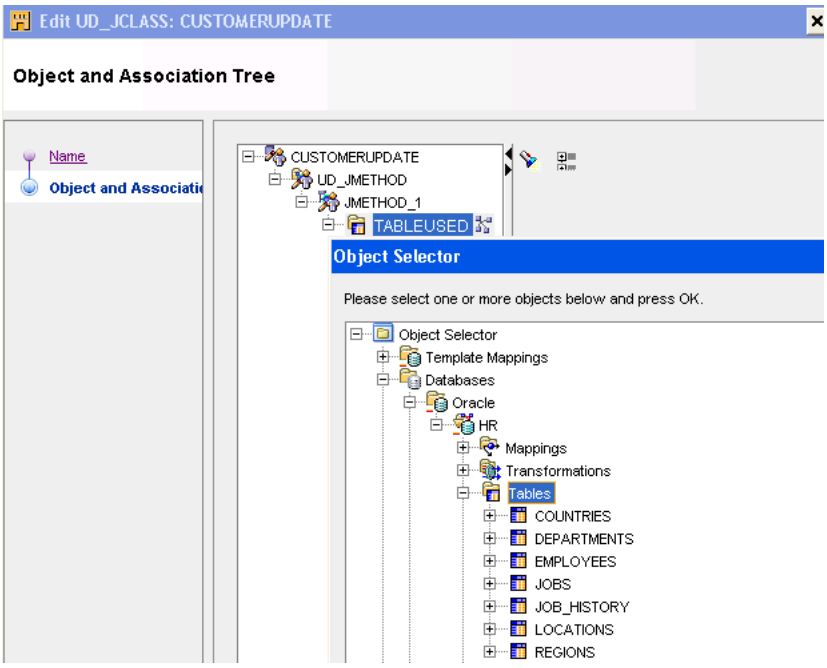
6. Right-click CUSTOMERUPDATE, and select **Open** to open the UDO Editor.
7. Click the Object and Association Tree tab and select CUSTOMERUPDATE. In the right panel, you can view the properties for the user defined object CUSTOMERUPDATE.



8. On the Object and Association Tree tab, right-click **UD_JMethod** and select **Create**.

   An SCO called JMETHOD_1 is created.
9. JMETHOD_1 contains two nodes: UD_JMETHOD_LINE, which is the child SCO, and TABLEUSED, which is the value specified for ROLE_1 when the association UD_XJMETHOD2TABLE was created.
10. Right-click TABLEUSED and select **Reference**.

    The Object Selector dialog box is displayed, and allows you to select the table to which you want to connect the UDO.



7.4 Working with UDOs and UDPs

In the graphical user interface, you can view UDOs and UDPs in the Projects Navigator and in the Repository Design Browser. However, in the Projects Navigator, you can also edit the UDOs and UDPs.

Repository Design Browser

The Repository Browser is a web browser for viewing repository metadata, generating different reports for objects metadata, and auditing runtime deployment and execution operations. In the Design Browser you can see all objects (including UDOs)and their properties (including UDPs) of all the projects created for a workspace. You can also generate and view reports on metadata including lineage and impact analysis reports.

If you define a UDP for a given object, then you can view the UDP name and values in the browser reports. The values are listed as Extended Property Name and Extended Property Value.

7.4.1 Propagating UDOs and UDPs to Other Workspaces

The primary method for propagating changes from one workspace to another is by using MDL. MDL enables you to export and import the metadata definition of the UDP and its contents.

Exporting UDOs and UDPs

You can export UDOs and UDPs as any other object.

In the MDL Control file, the option is `DEFINITIONFILE=`*`filename`* to export the metadata definition. For example:

```
## Sample Export file
USERID=UserName/Password@HostName:PortID:OracleServiceName
#
DEFINITIONFILE=Drive:\DirectoryName\filename.mdd
```

```
FILE=Drive:\DirectoryName\filename.mdl
LOG=Drive:\DirectoryName\filename.log
```

Importing UDOs and UDPs

You can import UDPs from the command line as well as from the user interface. During import, MDL updates the UDPs for all objects. In the MDL Control file, the option is DEFINITIONFILE=filename to import the metadata definition. For example:

```
## Sample Import file
USERID=UserName/Password@HostName:PortID:OracleServiceName
#
DEFINITIONFILE=Drive:\DirectoryName\filename.mdd

FILE=Drive:\DirectoryName\filename.mdl
LOG=Drive:\DirectoryName\filename.log
```

You can import UDPs using one of the following search criteria:

- **Universal ID**: The metadata definition contains a Universal Object ID (UOID). The UOID uniquely identifies objects across workspaces. If you import the MDL file by UOID, then MDL looks up the metadata definition by UOID. If the metadata definition name in the source MDL file is different from the metadata definition in the workspace, then MDL renames it when it is running in the Update/Replace mode.
- **Physical Name**: MDL looks up the metadata definition by physical name.

Regardless of the import mode, MDL either adds the metadata definition if it does not exist in the workspace, or updates the metadata definition if it already exists. MDL does not delete metadata definitions in the workspace.

When updating the metadata definition, MDL only renames the object if the names are different (search criteria is by UOID), and updates the default value. MDL does not change the data type.

7.5 Creating New Icons for Workspace Objects

Icons are graphics that visually suggest the availability of a function or type of an object to end users. There are many types of pre-defined workspace objects, each with their own icon. You may want to change the icon associated with an existing object or instance of an object to something more recognizable. For example, you could visually highlight a particular table by altering its icon. Additionally, for UDOs, you may want to change the default icon to something representative of the object. You can create your own icons using a graphics editor or third-party software.

You can create a new icon set and then associate it with an object using the OMB*Plus scripting language.

> **Note:** You can assign new icons to most workspace objects with the exception of pre-defined objects like public transformations and public data rules and `DEFAULT_CONFIGURATION`, `DEFAULT_CONTROL_CENTER`, and `OWB_REPOSITORY_LOCATION`.

Every object has a set of icons of varying sizes to represent it throughout the various editors and toolbars. Each icon set includes a canvas icon, palette icon, and a tree icon

as described in Table 7–1. When you define a new icon set, follow the sizing guidelines. If you specify a new icon with an incorrect size, it is automatically resized, which may distort your intended design.

Table 7–1 Icon Sets

Type	Description
Canvas Icon	Represents instances of objects in the canvas of an editor. For example, it displays a table icon in the canvas of the Mapping Editor or in a Lineage Report. The correct size is 32 x 32 pixels in GIF or JPEG format.
Palette Icon	Represents types of objects in editor palettes. For example, it displays the table operator in the Mapping Editor operator palette. The correct size is 18 x 18 pixels in GIF or JPEG format.
Tree Icon	Represents types and instances of objects in navigation trees such as the Project Explorer in the Design Center. The correct size is 16 x 16 pixels in GIF or JPEG format.

7.5.1 Creating Icon Sets

To create a new icon set, complete the following steps:

1. Log in to the client as an administrator.
2. In the Globals Navigator, right-click the Icon Sets node and select **New Icon Set**.
3. The Create Icon Set dialog box is displayed. For details on the values to be entered on this page, see "Create Icon Set Dialog Box" on page 7-13.

7.5.1.1 Create Icon Set Dialog Box

The Create Icon Set dialog box enables you to specify values for the Icon Set. Enter the following values and click **OK** to define a new Icon Set:

- **Name**: The name of the Icon Set.
- **Description**: A description of the Icon Set.
- **File Name**: Navigate and select the image that you want to assign to the new Icon Set. You need to select an image for Canvas, Palette, and Tree Icon.

> **Note:** You can select any image to represent the new icon.

The newly created icon set will be available under Icon Sets in the Globals Navigator.

7.5.2 Assigning New Icon Sets to Objects

To assign a new icon set to an object, you must use the OMB*Plus scripting clause SET REFERENCE ICONSET. To remove an icon set, use the clause UNSET REFERENCE ICONSET.

To assign an icon set named ICON1 to an FCO such as Table, use the following command:

```
OMBREDEFINE CLASS_DEFINITION 'TABLE' SET REF ICONSET 'ICON1'
```

To remove an icon set assigned to this FCO, use the following command:

```
OMBREDEFINE CLASS_DEFINITION 'TABLE' UNSET REF ICONSET
```

To assign an icon set to a specific instance of a table TAB1, use the following command:

```
OMBALTER TABLE 'TAB1' SET REFERENCE ICONSET 'ICON1'
```

To remove this icon set, use the following command:

```
OMBALTER TABLE 'TAB1' UNSET REFERENCE ICONSET
```

To know what icon set has been assigned to an object, use the following command:

```
OMBRETRIEVE TABLE 'TAB1' GET REFERENCE ICONSET
```

A

Additional and Optional Usages

This chapter includes the following topics:

- Using Control Files to Import and Export Metadata
- Working with Mappings and Operators
- Accessing Transformation Modules

A.1 Using Control Files to Import and Export Metadata

Control files enable you to specify additional options while importing or exporting metadata using the OMB*Plus commands OMBIMPORT and OMBEXPORT.

A.1.1 Creating MDL Control Files

An MDL control file is a text file that contains a set of parameters that are used by the Metadata Loader. These parameters specify the options to be used while exporting or importing metadata.

You can use any text editor to create a control file. Oracle recommends that you use the extension .ctl for control files. This helps identify it as an MDL control file.

The format for an export or import parameter is:

Keyword=Value

You can also form a parameter file by replacing the value with the wildcard character (*), which matches any string, or with a list of named objects:

Keyword=*

Keyword=(value-1, value-2, ..., -k)

For example, you can specify the keyword TABLES followed by the names of the tables to import as follows:

```
TABLES=(Customers, Products, Days)
```

You can use the comment indicator (#) to place comments in the control file. Put the comment indicator in the first column of a record and follow it with text.

Control File Example

An example of a control file is shown in Example A–1. This control file contains a list of object types (keywords) and their object names (values) to import from the MDL file. This is useful since the OMBIMPORT command does not provide an option to select the objects that you want to import from an MDL file.

Example A–1 Control File Format

```
PROJECT=MY_PROJECT
ORACLE_MODULES=DW1
TABLES=TABLE1
ORACLE_MODULES=DW2
DIMENSIONS=DIM1,DIM2
```

A.1.1.1 Keywords Used to Import Metadata

You use keywords to form the parameters specified to selectively import objects using a control file. Table A–1 provides a list of keywords for object types that you use to select objects to import.

Table A–1 Object Types Keywords for Importing Objects

Object Type Keyword	Description
ACTION	Used only for the Action Plan. The import mode must be MODE = ACTIONPLAN. The options are: ▪ CREATE ▪ REPLACE ▪ UPDATE ▪ INCREMENTALUPDATE (Merge mode) ▪ NONE (do not import the object) If a MODE parameter is not included, then the default is CREATE.
ACTIVITY_TEMPLATE_FOLDERS	Use this keyword to specify the activity template folders to be imported.
ACTIVITY_TEMPLATES	Use this keyword to specify the activity templates to be imported. Requires the ACTIVITY_TEMPLATE_FOLDERS to be specified.
ADVANCED_QUEUES	Use this keyword to specify the advanced queues to be imported. Requires the ORACLE_MODULES option to be specified.
BUSINESS_AREAS	Use this keyword to specify the business areas to be imported. Requires the BUSINESS_DEFINITION_MODULES option to be specified.
BUSINESS_DEFINITION_MODULES	Use this keyword to specify the business definition modules to be imported. Requires the PROJECT option to be specified.
BUSINESS_PRESENTATION_MODULES	Use this keyword to specify the business presentation modules to be imported. Requires the PROJECT option to be specified.
CALENDAR_MODULES	Use this keyword to specify the calendar modules to be imported.

Table A–1 (Cont.) Object Types Keywords for Importing Objects

Object Type Keyword	Description
CALENDARS	Use this keyword to specify the calendars to be imported. Requires the CALENDAR_MODULES option to be specified.
COLLECTIONS	Use this keyword to specify the collections to be imported. Requires the PROJECT option to be specified.
CONFIGURATIONS	Use this keyword to specify the configurations to be imported.
CONNECTORS	Use this keyword to specify the connectors to be imported. Requires the LOCATIONS option to be specified.
CONTROL_CENTERS	Use this keyword to specify the control centers to be imported. Requires the PROJECT option to be specified.
CUBES	Use this keyword to specify the cubes to be imported. Requires the ORACLE_MODULES option to be specified.
DATA_AUDITORS	Use this keyword to specify the data auditors to be imported. Requires the ORACLE_MODULES option to be specified.
DATA_RULE_MODULES	Use this keyword to specify the data rule modules to be imported. Requires the PROJECT option to be specified.
DATA_RULES	Use this keyword to specify the data rules to be imported. Requires the DATA_RULE_MODULES option to be specified.
DATA_PROFILES	Use this keyword to specify the data profiles to be imported. Requires the PROJECT option to be specified.
DEPLOYMENTS	Use this keyword to specify the deployments to be imported.
DIMENSIONS	Use this keyword to specify the dimensions to be imported. Requires the ORACLE_MODULES option to be specified.
DRILL_PATHS	Use this keyword to specify the drill paths to be imported. Requires the BUSINESS_DEFINITION_MODULES option to be specified.

Table A–1 (Cont.) Object Types Keywords for Importing Objects

Object Type Keyword	Description
EXPERT_MODULES	Use this keyword to specify the expert modules to be imported. Requires the PROJECT option to be specified.
EXPERTS	Use this keyword to specify the experts to be imported. Requires the EXPERT_MODULES option to be specified.
EXTERNAL_TABLES	Use this keyword to specify the external tables to be imported. Requires the ORACLE_MODULES option to be specified.
FLAT_FILE_MODULES	Use this keyword to specify the flat file modules to be imported. Requires the PROJECT option to be specified.
FLAT_FILES	Use this keyword to specify the flat files to be imported. Requires the FLAT_FILE_MODULES option to be specified.
FUNCTIONS	Use this keyword to specify the functions to be imported. Requires the module (for example, ORACLE_MODULES, BUSINESS_DEFINITION_MODULES, and so on) option to be specified.
GATEWAY_MODULES	Use this keyword to specify the Gateway modules to be imported. Requires the PROJECT option to be specified.
GENERIC_COMPONENTS	
GENERIC_FOLDERS	
GENERIC_MODULES	
ICONSETS	Use this keyword to specify the icon sets to be imported. Requires the ORACLE_MODULE option to be specified.
ITEM_CLASSES	Use this keyword to specify the item classes to be imported. Requires the BUSINESS_DEFINITION_MODULES option to be specified.
ITEM_FOLDERS	Use this keyword to specify the item folders to be imported. Requires the BUSINESS_DEFINITION_MODULES option to be specified.

Table A–1 (Cont.) Object Types Keywords for Importing Objects

Object Type Keyword	Description
LOCATIONS	Use this keyword to specify the locations to be imported. Requires the PROJECT option to be specified.
MAPPINGS	Use this keyword to specify the mappings to be imported. Requires the ORACLE_MODULES option to be specified.
MATERIALIZED_VIEWS	Use this keyword to specify the materialized views to be imported. Requires the ORACLE_MODULES option to be specified.
ORACLE_MODULES	Use this keyword to specify the Oracle modules to be imported. Requires the PROJECT option to be specified.
OBJECT_TYPES	Use this keyword to specify the object types to be imported. Requires the ORACLE_MODULES option to be specified.
PACKAGES	Use this keyword to specify the packages to be imported. Requires the ORACLE_MODULES option to be specified.
PLSSQL_RECORD_TYPES	Use this keyword to specify the PL/SQL record types to be imported.
PLSQL_REF_CURSOR_TYPES	Use this keyword to specify the PL/SQL REF cursor types to be imported.
PLSQL_TABLE_TYPES	Use this keyword to specify the PL/SQL types to be imported. Requires the ORACLE_MODULES option to be specified.
PLUGGABLE_MAPPING_FOLDERS	Use this keyword to specify the pluggable mapping folders to be imported.
PLUGGABLE_MAPPINGS	Use this keyword to specify the pluggable mappings to be imported. Requires the PLUGGABLE_ MAPPINGS_FOLDERS option to be specified.
PRESENTATION_TEMPLATES	Use this keyword to specify the presentation templates to be imported. Requires the BUSINESS_ PRESENTATION_MODULES option to be specified.

Table A–1 (Cont.) Object Types Keywords for Importing Objects

Object Type Keyword	Description
PROCESS_FLOW_MODULES	Use this keyword to specify the process flow modules to be imported. Requires the PROJECT option to be specified.
PROCESS_FLOW_PACKAGES	Use this keyword to specify the process flow packages to be imported. Requires the PROCESS_FLOW_ MODULES option to be specified.
PROCESS_FLOWS	Use this keyword to specify the process flows to be imported. Requires the PROCESS_FLOW_ PACKAGES option to be specified.
PROFILE_PREFERENCES	
PROJECT	Wildcard format supported, but if used, no other object type keywords can follow. To import shared transformations, use PROJECT=PUBLIC_PROJECT.
QUERYOBJECTS	
QUEUE_TABLES	Use this keyword to specify the queue tables to be imported. Requires the ADVANCED_QUEUES option to be specified.
ROLES	Use this keyword to import the roles to be imported. Requires administrator privileges.
SAP_MODULES	Use this keyword to specify the SAP modules to be imported. Requires the PROJECT option to be specified.
SEQUENCES	Use this keyword to specify the sequences to be imported. Requires the module (for example, ORACLE_MODULES, SAP_MODULES, and so on) options to be specified.
SHARED_MODULES	Requires PROJECT=PUBLIC_PROJECT option to be specified.
SNAPSHOTS	Use this keyword to specify the snapshots to be imported. If this option is used, the no other object type keyword options can precede it.
SQLCOLLECTIONS	
TABLES	Use this keyword to specify the tables to be imported. Requires the module (for example, ORACLE_MODULES, SAP_MODULES, and so on) options to be specified.

Table A–1 (Cont.) Object Types Keywords for Importing Objects

Object Type Keyword	Description
TRANSPORTABLE_MODULE_TABLESPACES	Use this keyword to specify the transportable module tablespace to be imported.
TRANSPORTABLE_MODULES	Use this keyword to specify the transportable modules to be imported.
USERS	Use this keyword to import the users to be imported. Requires administrator privileges.
VIEWS	Use this keyword to specify the views to be imported. Requires the module (for example, ORACLE_MODULES, SAP_MODULES, and so on) options to be specified.

A.1.1.2 Examples of Control Files Used to Import Metadata

You can direct the MDL import utility to import objects from a file by creating a control file with a set of parameters. Example A–2 shows a typical control file for importing objects from a MDL data file.

Example A–2 Control File format

```
PROJECT=WH_SALES
ORACLE_MODULES=SALES_SRC
EXTERNAL_TABLES=SALES_DATA_EXT
```

Control File Used to Create an Action Plan

You can specify an action plan in the control file that will allow you to specifically define what you want to do with each object in the imported file. First you need to indicate that the type of import is an action plan by specifying MODE = ACTION PLAN. Next, you need to specify the type of actions for objects that you want to import or skip. If you want to import the objects, you can set the Action to either CREATE, UPDATE, REPLACE, OR INCREMENTALUPDATE. Otherwise, if you want to skip the object, specify NONE as the Action. For the list of object type keywords that you can use to import objects using the Action Plan, refer Table A–1.

Example A–3 shows an example of an MDL control file that contains an action plan.

Example A–3 MDL Action Plan

```
MODE=ACTIONPLAN
#
# User-Specified Action Plan
#
ACTION=NONE
ORACLE_MODULES=(DATAWAREHOUSE)
#
ACTION=CREATE
TABLES=(TABLE_3)
FACTS=(FACT1, FACT2, FACT3)
SEQUENCES=(SEQ_A, SEQ_B, SEQ_C)
#
ACTION=REPLACE
```

```
TABLES=(TABLE_1, TABLE_2)
DIMENSIONS=(DIM1, DIM2, DIM3)
#
# Switching to a different module
ACTION=REPLACE
FLAT_FILE_MODULES=(FLAT_FILE)
FILES=(FILE_1, FILE_2)
#
ACTION=CREATE
FILES=(FILE_3)
#
```

A.1.2 Exporting Metadata Using OMB*Plus

You use the OMBEXPORT command to export metadata. To use control files in conjunction with the OMBEXPORT command, use the CONTROL_FILE clause. For more information on creating control files, see "Creating MDL Control Files" on page A-1.

Before you use the OMBEXPORT command, ensure that you are connected to the repository from which you want to export metadata. You use the OMBCONNECT command to connect to a repository. For more information on the OMBCONNECT and OMBEXPORT commands, refer to the OMBEXPORT command in the Oracle Warehouse Builder API and Scripting Reference.

Examples of Exporting Metadata Using Control Files

To use control files in conjunction with the OMBEXPORT command:

1. Create an MDL control file.

   For more information on creating an MDL control file, see "Creating MDL Control Files" on page A-1.

2. Open OMB*Plus by first selecting **Start**, then **Programs**, then **<OWB Home>**, then **Warehouse Builder**, and, finally, **OMB Plus**.

3. Connect to the repository from which you want to export metadata.

4. Execute the OMBEXPORT command with the CONTOL_FILE clause that specifies the control file created in Step 1.

For example, to use a control file called `par.ctl` use the following OMBEXPORT command:

```
OMBEXPORT TO MDL_FILE 'd:/mdl/exp1.mdl' FROM PROJECT 'MY_PROJECT'
CONTROL_FILE 'd:/mdl/par.ctl' OUTPUT LOG TO 'd:/mdl/exp1.log'
```

A.1.3 Importing Metadata Using OMB*Plus

You use the OMBIMPORT command to import metadata. You can also use control files in conjunction with the OMBIMPORT command. To do this, use the CONTROL_FILE clause of the OMBIMPORT command. For more information on creating control files, see "Creating MDL Control Files" on page A-1.

Before you use the OMBIMPORT command, ensure that you are connected to the repository into which you want to import metadata. You use the OMBCONNECT command to connect to a repository. For more information on the OMBCONNECT and OMBIMPORT commands, refer to Oracle Warehouse Builder API and Scripting Reference.

Examples of Importing Metadata Using Control Files

To use control files in conjunction with the OMBIMPORT command:

1. Create an MDL control file.

   For more information on creating an MDL control file, see "Creating MDL Control Files" on page A-1.

2. Open OMB*Plus by first selecting **Start**, then **Programs**, then **<OWB Home>**, then **Warehouse Builder**, and, finally, **OMB Plus**

3. Connect to the repository from which you want to import metadata.

4. Execute the OMBIMPORT command with the CONTOL_FILE clause that specifies the control file created in Step 1.

For example, to use a control file called `par.ctl` use the following OMBIMPORT command:

```
OMBIMPORT FROM MDL_FILE 'd:/mdl/exp1.mdl'

CONTROL_FILE 'd:/mdl/par.ctl' OUTPUT LOG TO 'd:/mdl/exp1.log'
```

A.1.4 Accessing Transformation Modules Using OMB*Plus

Transformation modules consist of a set of reusable transformations that you use to transform your source data. Transformations include functions, procedures, and packages.

There are two types of transformation modules:

- Predefined Transformations
- Custom Transformations

A.1.4.1 Predefined Transformations

Predefined transformations consist of built-in and seeded functions and procedures that are part of the Oracle Library. You can directly use these transformations in any project in your repository.

Predefined transformations are grouped into the following categories. Each category contains transformations that pertain to that category.

- Administration
- Character
- Control Center
- Conversion
- Date
- Numeric
- OLAP
- Other
- Spatial
- Streams
- SYS
- XML

A.1.4.2 Accessing Predefined Transformations Using OMB*Plus

All predefined transformations belong to a transformation module called WB_PREDEFINED_TRANS in the project PUBLIC_PROJECT. Also, every project in the repository contains the WB_PREDEFINED_TRANS. To access predefined transformations, you must change the current context to the WB_PREDEFINED_TRANS transformation module either in PUBLIC_PROJECT or in your project.

Each category of predefined transformations is represented by a package in the WB_PREDEFINED_TRANS transformation module. The package contains the transformations, including functions and procedures, that are belong under it. For example, all the predefined numeric transformations belong to the package called NUMERIC under the WB_PREDEFINED_TRANS transformation module.

Examples

Use the following commands to list the types of public transformation modules.

```
OMB+> OMBCC '/PUBLIC_PROJECT/'
OMB+> OMBLIST TRANSFORMATION_MODULES
```

To view the types of predefined transformations, use the following command from the context of the WB_PREDEFINED_TRANS transformation module.

```
OMB+> OMBCC '/PUBLIC_PROJECT/WB_PREDEFINED_TRANS'
OMB+> OMBLIST PACKAGES
```

To view the procedures under the Date category of the predefined transformations, first change context to the DATE package.

```
OMB+> OMBCC '/PUBLIC_PROJECT/WB_PREDEFINED_TRANS/PACKAGES/DATE'
OMB+> OMBLIST PROCEDURES
```

Use the following command to use the Date transformation TRUNC in your mapping.

```
OMB+> OMBCREATE MAPPING 'MAP1'\
    > ADD TRANSFORMATION OPERATOR 'TRUNC_OPER' \
    > BOUND TO FUNCTION '/MY_PROJECT/WB_PREDEFINED_TRANS/DATE/TRUNC'
```

A.1.4.3 Custom Transformations

Custom transformations are transformations that are defined by the user. Custom transformations include functions, procedures, and packages.

Custom transformations are of two types:

- Public custom transformations

  These are part of the global shared library that consists of predefined transformations.

- Custom transformations within a particular project

  These are accessible only in the project in which they are defined.

A.1.4.4 Public Custom Transformations

Public custom transformations are accessible across all projects in your repository. They belong to the transformation module WB_CUSTOM_TRANS under the project PUBLIC_PROJECT. Also, every project in your repository contains a transformation module called WB_CUSTOM_TRANS. This transformation module contains the public custom transformations.

Examples

To list the types of custom transformations, you need to be in the context of the custom transformations module.

```
OMB+> OMBCC '/PUBLIC_PROJECT/WB_CUSTOM_TRANS'
OMB+> OMBLIST FUNCTIONS
```

To view the custom public procedures, use the following command from the context of the WB_CUSTOM_TRANS module in the PUBLIC_PROJECT.

```
OMB+> OMBLIST PACKAGES
```

To use a public custom function in a mapping, navigate to the context of the WB_CUSTOM_TRANS transformation module under the project in which you are defining the mapping.

```
OMB+> OMBCC '/MY_PROJECT/MOD1'
OMB+> OMBCREATE MAPPING 'MAP1'\
    > ADD TRANSFORMATION OPERATOR 'FUNC_OPER' \
    > BOUND TO FUNCTION '/PUBLIC_PROJECT/WB_CUSTOM_TRANS/FUNC1
```

A.1.4.5 Custom Transformations that Belong to a Particular Project

You can create custom transformations whose scope is limited to the project in which they are defined. These custom transformations are defined in the context of a particular project and are accessible to all the modules within that project.

For example, the project MY_PROJECT contains two modules MOD1 and MOD2. In MOD1, you define a function called LOCAL_FUNC. This function is accessible from the context of both MOD1 and MOD2.

Custom transformations that belong to a particular project are part of the transformations in that project.

To create a custom transformation in the module MOD1, use the following syntax.

```
OMB+> OMBCREATE FUNCTION 'LOCAL_FUNC'\
    > ADD PARAMETER PARAM_1\
    > SET PROPERTIES (IN_OUT,DATATYPE) VALUES('in','varchar2')
```

Use the following command to reference the function LOCAL_FUNC in a mapping that you defined in module MOD2.

```
OMB+> OMBCREATE MAPPING 'MAP1'\
    > ADD TRANSFORMATION OPERATOR 'TRUNC_OPER' \
    > BOUND TO FUNCTION '/MY_PROJECT/MOD1/LOCAL_FUNC'
```

A.2 Working with Mappings and Operators

This section includes the following topics for using OMB*Plus to add operators to mappings:

A.2.1 Defining Expressions in Mappings

When using the OMBCREATE MAPPING and OMBALTER MAPPING commands, you can create and edit expressions such as filter, join, and group by expressions. If you define the attributes of the operator before defining the expression, OMB*Plus

generates the expected code. However, if you define the expression incorrectly and without first defining the necessary attributes, OMB*Plus interprets your entry as a string literal and generates unexpected code.

To correctly define an expression in a mapping, prefix each attribute name with a colon. OMB*Plus recognizes the text following a colon as an attribute. For example, OMB*Plus interprets

:INOUTGRP1.ATTR1

as an attribute `ATTR1` in a group named `INOUTGRP1`.

If you make it a practice when writing expressions to precede attribute names with a colon, OMB*Plus returns an error message in the event that you defined the expression without first defining the operator attributes.

A.2.2 Default Group Names and Attribute Names

When you use OMB*Plus to add an operator to a mapping, Warehouse Builder adds the operator and assigns default groups and parameters. Table A–2 lists the default groups and parameters for each operator.

Table A–2 Default Names for Groups and Attributes

Operator Type	Default Operator Name	Default Group Name	Default Attribute Name
TABLE VIEW EXTERNAL TABLE MATERIALIZED VIEW CUBE DIMENSION	Same as bound object name	INOUTGRP1	Same as column names
FLAT FILE	Object name	Same as file record name	Same as field names in each record
SEQUENCE	Same as sequence name	OUTGRP1	NEXTVAL CURRVAL
DATAGENERATOR	DATAGENERATOR	OUTGRP1	RECNUM SYS_DATE SEQUENCE
CONSTANT	CONST	OUTGRP1	No defaults
KEY LOOKUP	Object name	INGRP1 OUTGRP1	In LOOKUP_OUT object column names
SET	SET	INGRP1 INGRP2 OUTGRP1	None
JOINER	JOIN	INGRP1 INGRP2 OUTGRP1	None

Table A–2 (Cont.) Default Names for Groups and Attributes

Operator Type	Default Operator Name	Default Group Name	Default Attribute Name
SPLITTER	SPLIT	INGRP1 OUTGRP1 OUTGRP2 REMAINING_ ROWS	None
DEDUPLICATOR	DEDUP	INOUTGRP1	None
AGGREAGTOR	AGG	INGRP1 OUTGRP1	None
FILTER	FLTR	INOUTGRP1	None
SORTER	SORT	INOUTGRP1	None
NAME AND ADDRESS	NAMEADDR	INGRP1 OUTGRP1	Within Group INGRP1: Line 1, Line 2, Line 3 Within Group OUTGRP1: First Name, Last Name, Primary Address, Secondary Address, City, State, Postal Code, Is Good Group
PROCEDURES	Procedure name	If input parameter exists, an input group will be created with the name INGRP1. If output parameter exists, an output group will be created with the name OUTGRP1. If inout parameter exists, an input-output group will be created with the name INGRP1.	Same as parameter names
FUNCTIONS	Same as function name	INGRP1, RETURN	An attribute
TRIGGER	PREMAP POSTMAP	Naming see PROCEDURES and FUNCTIONS	None
INPUT_PARAMETER	MAP_INPUTS	MAP_INPUTS	None
OUTPUT_PARAMETER	MAP_OUTPUTS	MAP_OUTPUTS	None
EXTERNAL_PROCESS	EXTERNALPROCESS	None	None
EXPRESSION	EXPR	INGRP1 OUTGRP1	None

A.3 Accessing Transformation Modules

Transformation modules consist of a set of reusable transformations that you use to transform your source data. Transformations include functions, procedures, and packages.

There are two types of transformation modules:

- Predefined Transformations
- Custom Transformations

A.3.1 Predefined Transformations

Predefined transformations consist of built-in and seeded functions and procedures that are part of the Oracle Library. You can directly use these transformations in any project in your repository.

Predefined transformations are grouped into the following categories. Each category contains transformations that pertain to that category.

- Administration
- Character
- Control Center
- Conversion
- Date
- Numeric
- OLAP
- Other
- Spatial
- Streams
- SYS
- XML

A.3.1.1 Accessing Predefined Transformations Using OMB*Plus

All predefined transformations belong to a transformation module called `WB_PREDEFINED_TRANS` in the project `PUBLIC_PROJECT`. Also, every project in the repository contains the `WB_PREDEFINED_TRANS`. To access predefined transformations, you must change the current context to the `WB_PREDEFINED_TRANS` transformation module either in `PUBLIC_PROJECT` or in your project.

Each category of predefined transformations is represented by a package in the `WB_PREDEFINED_TRANS` transformation module. The package contains the transformations, including functions and procedures, that are belong under it. For example, all the predefined numeric transformations belong to the package called `NUMERIC` under the `WB_PREDEFINED_TRANS` transformation module.

Examples

Use the following commands to list the types of public transformation modules.

```
OMB+> OMBCC '/PUBLIC_PROJECT/'
OMB+> OMBLIST TRANSFORMATION_MODULES
```

To view the types of predefined transformations, use the following command from the context of the WB_PREDEFINED_TRANS transformation module.

```
OMB+> OMBCC '/PUBLIC_PROJECT/WB_PREDEFINED_TRANS'
OMB+> OMBLIST PACKAGES
```

To view the procedures under the Date category of the predefined transformations, first change context to the DATE package.

```
OMB+> OMBCC '/PUBLIC_PROJECT/WB_PREDEFINED_TRANS/PACKAGES/DATE'
OMB+> OMBLIST PROCEDURES
```

Use the following command to use the Date transformation TRUNC in your mapping.

```
OMB+> OMBCREATE MAPPING 'MAP1'\
    > ADD TRANSFORMATION OPERATOR 'TRUNC_OPER' \
    > BOUND TO FUNCTION '/MY_PROJECT/WB_PREDEFINED_TRANS/DATE/TRUNC'
```

A.3.2 Custom Transformations

Custom transformations are transformations that are defined by the user. Custom transformations include functions, procedures, and packages.

Custom transformations are of two types:

- Public custom transformations

  These are part of the global shared library that consists of predefined transformations.

- Custom transformations within a particular project

  These are accessible only in the project in which they are defined.

A.3.2.1 Public Custom Transformations

Public custom transformations are accessible across all projects in your repository. They belong to the transformation module WB_CUSTOM_TRANS under the project PUBLIC_PROJECT. Also, every project in your repository contains a transformation module called WB_CUSTOM_TRANS. This transformation module contains the public custom transformations.

Examples

To list the types of custom transformations, you need to be in the context of the custom transformations module.

```
OMB+> OMBCC '/PUBLIC_PROJECT/WB_CUSTOM_TRANS'
OMB+> OMBLIST FUNCTIONS
```

To view the custom public procedures, use the following command from the context of the WB_CUSTOM_TRANS module in the PUBLIC_PROJECT.

```
OMB+> OMBLIST PACKAGES
```

To use a public custom function in a mapping, navigate to the context of the WB_CUSTOM_TRANS transformation module under the project in which you are defining the mapping.

```
OMB+> OMBCC '/MY_PROJECT/MOD1'
OMB+> OMBCREATE MAPPING 'MAP1'\
    > ADD TRANSFORMATION OPERATOR 'FUNC_OPER' \
    > BOUND TO FUNCTION '/PUBLIC_PROJECT/WB_CUSTOM_TRANS/FUNC1
```

A.3.2.2 Custom Transformations that Belong to a Particular Project

You can create custom transformations whose scope is limited to the project in which they are defined. These custom transformations are defined in the context of a particular project and are accessible to all the modules within that project.

For example, the project MY_PROJECT contains two modules MOD1 and MOD2. In MOD1, you define a function called LOCAL_FUNC. This function is accessible from the context of both MOD1 and MOD2.

Custom transformations that belong to a particular project are part of the transformations in that project.

To create a custom transformation in the module MOD1, use the following syntax.

```
OMB+> OMBCREATE FUNCTION 'LOCAL_FUNC'\
    > ADD PARAMETER PARAM_1\
    > SET PROPERTIES (IN_OUT,DATATYPE) VALUES('in','varchar2')
```

Use the following command to reference the function LOCAL_FUNC in a mapping that you defined in module MOD2.

```
OMB+> OMBCREATE MAPPING 'MAP1'\
    > ADD TRANSFORMATION OPERATOR 'TRUNC_OPER' \
    > BOUND TO FUNCTION '/MY_PROJECT/MOD1/LOCAL_FUNC'
```

Index

P

R

S

T

U

V

W