Oracle® Data Miner
Installation and Administration Guide
Release 4.1
E58244-05

June 2015
Contents

Preface ............................................................................................................................................................ vii
Audience ...................................................................................................................................................... vii
Documentation Accessibility .................................................................................................................... vii
Related Documents ................................................................................................................................. vii
Conventions ............................................................................................................................................... viii

1 Oracle Data Miner System Overview
   1.1 Oracle Data Miner Architecture................................................................................................. 1-1
   1.2 About the Oracle Data Miner Repository.................................................................................... 1-2
   1.3 Database Features Used by Oracle Data Miner.......................................................................... 1-2
   1.4 Oracle Data Miner and Oracle Advanced Analytics ................................................................. 1-3
   1.5 About the Data Mining APIs....................................................................................................... 1-3
       1.5.1 Data Mining PL/SQL Packages.......................................................................................... 1-4
       1.5.2 Data Mining SQL Scoring Functions ................................................................................. 1-4
       1.5.3 Data Mining Data Dictionary Views.................................................................................... 1-5
   1.6 Resources For Learning About Oracle Data Miner....................................................................... 1-5

2 Installing the Database to Support Oracle Data Miner
   2.1 Database Requirements for Oracle Data Miner........................................................................... 2-1
   2.2 Oracle Text and Oracle Data Miner............................................................................................. 2-2
   2.3 XML DB and Oracle Data Miner.................................................................................................. 2-2
   2.4 Storage Configuration for Oracle Data Miner............................................................................. 2-3
   2.5 Installing Oracle Database 11.2 or 12.1....................................................................................... 2-3

3 Installing Oracle Data Miner
   3.1 Oracle Data Miner Installation Overview ..................................................................................... 3-1
   3.2 Downloading SQL Developer....................................................................................................... 3-2
   3.3 Installing the Repository Using a Script....................................................................................... 3-2
   3.4 Loading the Sample Data Using a Script..................................................................................... 3-3
       3.4.1 About the Oracle Data Miner Sample Data ........................................................................ 3-4
   3.5 Installing the Repository Using the SQL Developer GUI......................................................... 3-4
   3.6 Installing the JSON Parser and Data Guide................................................................................. 3-6
       3.6.1 About the Scripts for Installing JSON Support ................................................................. 3-6
       3.6.2 Requirements for Running the Scripts............................................................................... 3-6
       3.6.3 Example: Installing JSON Support in a Remote Database................................................ 3-6
4 Managing Oracle Data Miner Users

4.1 About User Objects and Repository Objects ................................................................. 4-1
4.1.1 Objects in the Schema of the Oracle Data Miner User .............................................. 4-1
4.1.2 About Oracle Data Miner Internal Tables ................................................................. 4-1
4.2 About Proxy Users for Oracle Data Miner ................................................................. 4-2
4.3 Choosing an Access Model for Oracle Data Miner ..................................................... 4-3
4.3.1 About Shared Access ................................................................................................. 4-4
4.3.1.1 About the Document in Use Condition ............................................................... 4-4
4.3.2 About Single User Access ......................................................................................... 4-4
4.4 Granting or Dropping Access Rights to the Oracle Data Miner Repository ............... 4-5
4.4.1 Granting Access Rights Using the GUI .................................................................... 4-5
4.4.2 Granting Access Rights Using a Script .................................................................... 4-5
4.4.3 Dropping Access Rights Using a Script .................................................................... 4-5
4.5 Granting Access to Data ............................................................................................... 4-6

5 Managing the Oracle Data Miner Repository

5.1 About the Oracle Data Miner Administration Scripts .................................................. 5-1
5.1.1 Setting the Path to the Oracle Data Miner Scripts ................................................... 5-1
5.2 Determining the Status and Version of the Repository ............................................... 5-1
5.3 Backing Up and Restoring the Repository .................................................................. 5-2
5.3.1 Full Backup and Restore ......................................................................................... 5-2
5.3.1.1 Full Backup and Restore in Database 11.2.0.1 to 11.2.0.3 ................................... 5-2
5.3.1.2 Full Backup and Restore in Database 11.2.0.4 and Higher ................................ 5-2
5.3.2 Workflow Only Backup ......................................................................................... 5-3
5.3.3 Workflow Only Restore ......................................................................................... 5-3
5.3.4 Workflow Only Restore Examples ......................................................................... 5-4
5.4 Migrating the Repository ............................................................................................ 5-5
5.4.1 Upgrade ODMRSYS ............................................................................................... 5-6
5.4.2 Upgrade New ODMRSYS Tablespace From Object-Relational to Binary ............. 5-6
5.4.3 Upgrade ODMRSYS From Object-Relational to Binary ....................................... 5-7
5.4.4 Upgrade Binary Workflow Data in ODMRSYS ....................................................... 5-7
5.5 Dropping the Repository ............................................................................................ 5-8

6 Managing System Resources for Oracle Data Miner

6.1 Oracle Data Miner Resource Management Overview .................................................. 6-1
6.2 Allocating Resources to Oracle Data Miner User Sessions ........................................ 6-1
6.3 Managing Model Builds ............................................................................................. 6-3
6.4 Managing Workflow Execution ................................................................................... 6-4
6.5 Managing Parallel Execution ...................................................................................... 6-5
6.6 Summary of Oracle Data Miner Repository Properties for System Management ....... 6-5

7 Generating and Deploying SQL Scripts

7.1 Overview of the SQL Script Generation Feature ......................................................... 7-1
7.1.1 Audience ................................................................................................................ 7-1
8 Using PL/SQL API to Manage Workflows

8.1 About PL/SQL APIs ................................................................. 8-1
8.2 PL/SQL APIs ........................................................................... 8-1
8.2.1 PROJECT_CREATE .............................................................. 8-2
8.2.2 PROJECT_RENAME .............................................................. 8-2
8.2.3 PROJECT_DELETE ............................................................... 8-2
8.2.4 WF_RUN .............................................................................. 8-3
8.2.4.1 WF_RUN with Project Name, Workflow Name, and Node Name .... 8-4
8.2.4.2 WF_RUN with Project ID, Workflow ID and Node IDs ................. 8-4
8.2.4.3 WF_RUN with Project Name, Workflow Name Node Name and Time Interval ...... 8-4
8.2.4.4 WF_RUN with Project ID, Workflow ID, Node ID and Time Interval ...... 8-5
8.2.5 WF_STOP ........................................................................... 8-5
8.2.6 WF_RENAME .................................................................... 8-5
8.2.7 WF_DELETE ..................................................................... 8-6
8.2.8 WF_IMPORT ..................................................................... 8-6
8.2.9 WF_EXPORT ..................................................................... 8-7
8.3 Repository Views .................................................................. 8-7
8.3.1 ODMR_USER_PROJECT_WORKFLOW ...................................... 8-7
8.3.2 ODMR_USER_WORKFLOW_ALL ........................................... 8-8
8.3.3 ODMR_USER_WORKFLOW_LOG ......................................... 8-9
8.3.4 ODMR_USER_WORKFLOW_NODES ...................................... 8-11
8.3.5 ODMR_USER_WORKFLOW_MODELS .................................... 8-12
8.3.6 ODMR_USERWF_CLAS_TEST_RESULTS .............................. 8-13
8.3.7 ODMR_USERWF_REGR_TEST_RESULTS .............................. 8-15
<table>
<thead>
<tr>
<th>Section</th>
<th>Main Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.3.8</td>
<td>ODMR_USER_WF_TEST_RESULTS</td>
<td>8-16</td>
</tr>
<tr>
<td>8.4</td>
<td>PL/SQL APIs Use Cases</td>
<td>8-18</td>
</tr>
<tr>
<td>8.4.1</td>
<td>Premise of the Use Case</td>
<td>8-18</td>
</tr>
<tr>
<td>8.4.2</td>
<td>Location of Demo Workflow Files</td>
<td>8-18</td>
</tr>
<tr>
<td>8.4.3</td>
<td>Use Case to Schedule and Run a Build Workflow</td>
<td>8-18</td>
</tr>
<tr>
<td>8.4.3.1</td>
<td>Querying Named Objects</td>
<td>8-20</td>
</tr>
<tr>
<td>8.4.3.2</td>
<td>Querying the MODEL_COEFFICIENT Table</td>
<td>8-20</td>
</tr>
<tr>
<td>8.4.3.3</td>
<td>Querying Test Results</td>
<td>8-20</td>
</tr>
<tr>
<td>8.4.4</td>
<td>Use case to Schedule and Run an Apply Workflow</td>
<td>8-22</td>
</tr>
<tr>
<td>8.4.4.1</td>
<td>Querying Scoring Result</td>
<td>8-23</td>
</tr>
</tbody>
</table>

### A Oracle Data Miner Releases

**Index**
Preface

*Oracle Data Miner Installation and Administration Guide* explains how to use the Data Miner scripts that are included with SQL Developer to install and administer the Data Miner repository in Oracle Database.

**Audience**

This document is intended for database administrators and database developers.

**Documentation Accessibility**


**Access to Oracle Support**

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

**Related Documents**

For more information, see the following documents in the Oracle Database Online Documentation Library:

**Oracle Database 12c**

- Oracle Data Miner Release Notes
- Oracle Data Miner User’s Guide
- Oracle SQL Developer User’s Guide
- Oracle Data Mining Concepts
- Oracle Data Mining Application Developer’s Guide
- Oracle Data Mining API Guide (Virtual Book)
- Oracle R Enterprise User’s Guide
- Oracle R Enterprise Installation and Administration Guide

**Oracle Database 11g**
Conventions

The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td><strong>monospace</strong></td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
This chapter introduces Oracle Data Miner and the programmatic interfaces of Oracle Data Mining. It also supplies links to resources to help you learn more about the product. This chapter contains these topics:

- Oracle Data Miner Architecture
- About the Oracle Data Miner Repository
- Database Features Used by Oracle Data Miner
- Oracle Data Miner and Oracle Advanced Analytics
- About the Data Mining APIs
- Resources For Learning About Oracle Data Miner

1.1 Oracle Data Miner Architecture

Oracle Data Miner is an extension of Oracle SQL Developer, a graphical development environment for Oracle SQL. Oracle Data Miner uses the data mining technology embedded in Oracle Database to create, execute, and manage workflows that encapsulate data mining operations. Oracle Data Miner uses the ODMRSYS schema as a dedicated system repository.

The architecture of Oracle Data Miner is illustrated in Figure 1–1.
1.2 About the Oracle Data Miner Repository

Oracle Data Miner requires the installation of a repository, the ODMRSYS schema, in the database server. Oracle Data Miner users must have the privileges that are required for accessing objects in ODMRSYS.

The Oracle Data Miner repository manages:

- **Storage**: The repository stores the projects and workflows of all the Oracle Data Miner users that have established connections to this database.
- **Runtime Functions**: The repository is the application layer for Oracle Data Miner. It controls the execution of workflows and other runtime operations.

1.3 Database Features Used by Oracle Data Miner

Oracle Data Miner uses these features of Oracle Database:

- **Oracle Data Mining**: Provides the model building, testing, and scoring capabilities of Oracle Data Miner.
- **Oracle XML DB**: Manages the metadata in the Oracle Data Miner repository.
- **Oracle Text**: Supports text mining.
- **Oracle Scheduler**: Schedules workflow execution.
- **Oracle R Enterprise**: Executes embedded R scripts supplied by the user.
Oracle Data Miner System Overview

1.4 Oracle Data Miner and Oracle Advanced Analytics

Oracle Data Miner is the Graphical User Interface (GUI) for Oracle Data Mining, the data mining engine in Oracle Database. Oracle Data Mining is a component of the Oracle Advanced Analytics option of Oracle Database Enterprise Edition.

Components of Oracle Advanced Analytics:

- **Oracle Data Mining** (required by Oracle Data Miner)

  Oracle Data Mining is a powerful data mining engine embedded in the Database kernel. Oracle Data Mining supports algorithms for classification, regression, clustering, feature selection, feature extraction, and association (market basket analysis). The Data Mining PL/SQL Application Programming Interface (API) performs data preparation and creates, evaluates, and maintains mining models. Data Mining SQL functions score data using mining models or predictive queries.

- **Oracle R Enterprise** (not required by Oracle Data Miner)

  Oracle Data Miner provides limited support for Oracle R Enterprise. If a user supplies a script that includes embedded R in the Oracle Data Miner SQL Query node, then Oracle Data Miner uses Oracle R Enterprise to execute the script.

  Oracle R Enterprise integrates the open source R statistical programming language and environment with Oracle Database. Oracle R Enterprise supports a transparency layer, which allows R to act transparently on Oracle data, and embedded R execution, which allows the execution of R scripts in the database.

  **See Also:** Oracle Data Mining Concepts

  Oracle R Enterprise User’s Guide

1.5 About the Data Mining APIs

Oracle Data Miner is an application based on the Data Mining APIs in Oracle Database. The APIs are public and can also be used directly for application development. The APIs are summarized in the following topics:

- Data Mining PL/SQL Packages
- Data Mining SQL Scoring Functions

Note: With the exception of Oracle R Enterprise, these features are all included by default in Oracle Database Enterprise Edition. Oracle R Enterprise requires additional installation steps, as described in *Oracle R Enterprise Installation and Administration Guide*.

See Also: The following documents for details about the features used by Oracle Data Miner:

- Oracle Data Mining Concepts
- Oracle Data Mining Application Developer’s Guide
- Oracle XML DB Developer’s Guide
- Oracle Text Application Developer’s Guide
- Oracle Database Administrator’s Guide
- Oracle R Enterprise User’s Guide
### Data Mining Data Dictionary Views

#### 1.5.1 Data Mining PL/SQL Packages

PL/SQL language APIs manipulate mining models, which are database schema objects. Table 1–1 lists the PL/SQL packages and their descriptions.

<p>| Table 1–1 Oracle Data Mining PL/SQL Packages |</p>
<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBMS_DATA_MINING</td>
<td>DDL procedures for managing mining models. Mining model settings. Procedures for testing mining models, functions for querying mining models, and an APPLY procedure for batch scoring.</td>
</tr>
<tr>
<td>DBMS_DATA_MINING_TRANSFORM</td>
<td>Procedures for specifying transformation expressions and applying the transformations to columns of data. Transformations can be passed to the model creation process and embedded in the model definition, or they can be applied externally to data views.</td>
</tr>
<tr>
<td>DBMS_PREDICTIVE_ANALYTICS</td>
<td>Procedures that perform predict, explain, and profile operations without a user-created mining model.</td>
</tr>
</tbody>
</table>

---

**Note:** The mining operations in the DBMS_PREDICTIVE_ANALYTICS package are available in code snippets in Oracle Data Miner, as described in Oracle Data Miner User’s Guide.

---

**See Also:** Oracle Database 12.1: Oracle Database PL/SQL Packages and Types Reference

Oracle Database 11.2: Oracle Database PL/SQL Packages and Types Reference

#### 1.5.2 Data Mining SQL Scoring Functions

A set of specialized SQL functions provides the primary mechanism for scoring data in Oracle Data Mining. When called as single-row functions, the SQL Data Mining functions apply a user-supplied mining model to each row of input data. In Oracle Database 12c, the functions can also be called as analytic functions, in which case the algorithmic processing is performed dynamically without a user-supplied mining model. The term *predictive query* is refers to this mode of scoring.

<p>| Table 1–2 Data Mining SQL Scoring Functions |</p>
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Function Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLUSTER_DETAILS</td>
<td>Returns cluster details for each row in the input data.</td>
</tr>
<tr>
<td>CLUSTER_DISTANCE</td>
<td>Returns the distance between each row and the centroid.</td>
</tr>
<tr>
<td>CLUSTER_ID</td>
<td>Returns the ID of the highest probability cluster for each row.</td>
</tr>
<tr>
<td>CLUSTER_PROBABILITY</td>
<td>Returns the highest probability cluster for each row.</td>
</tr>
<tr>
<td>CLUSTER_SET</td>
<td>Returns a set of cluster ID and probability pairs for each row.</td>
</tr>
<tr>
<td>FEATURE_DETAILS</td>
<td>Returns a set of feature and value paris for each row.</td>
</tr>
<tr>
<td>FEATURE_ID</td>
<td>Returns feature details for each row in the input data.</td>
</tr>
</tbody>
</table>
### 1.5.3 Data Mining Data Dictionary Views

The data dictionary views store information about mining models in the Oracle Database system catalog. All views are available for DBA, USER, and ALL access. Table 1–3 lists the Data mining data dictionary views and their descriptions.

<table>
<thead>
<tr>
<th>View Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*_MINING_MODELS</td>
<td>Provides information about all accessible mining models</td>
</tr>
<tr>
<td>*_MINING_MODEL_ATTRIBUTES</td>
<td>Provides information about the attributes of all accessible mining models</td>
</tr>
<tr>
<td>*_MINING_MODEL_SETTINGS</td>
<td>Provides information about the settings of all accessible mining models</td>
</tr>
</tbody>
</table>

**Note:** The SQL scoring functions are available in code snippets in Oracle Data Miner, as described in Oracle Data Miner User’s Guide.

**See Also:** Oracle Database 12.1:
- Oracle Database SQL Language Reference
- Oracle Data Mining Application Developer’s Guide

Oracle Database 11.2:
- Oracle Database SQL Language Reference
- Oracle Data Mining Application Developer’s Guide
- Oracle Data Mining Application Developer’s Guide

### 1.6 Resources For Learning About Oracle Data Miner

Table 1–1 lists the Oracle Data Miner documentation and training resources.
<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Oracle Data Miner Documentation** | Oracle Data Miner User’s Guide  
Oracle Data Miner Release Notes  
Oracle Data Miner online help |
| **Oracle Data Mining 12.1 Documentation** | Oracle Data Mining Concepts  
Oracle Data Mining Application Developer’s Guide  
Oracle Data Mining API Guide (Virtual Book) |
| **Oracle Data Mining 11.2 Documentation** | Oracle Data Mining Concepts  
Oracle Data Mining Application Developer’s Guide  
Oracle Data Mining API Guide (Virtual Book) |
| Tutorials | Oracle Data Mining 4.0 OBE (Oracle By Example) Series  
Text Mining Using Oracle Data Miner 3.0  
Star Schema Mining Using Oracle Data Miner 3.0  
| Forum | https://community.oracle.com/community/business_intelligence/data_warehousing/data_mining |
| Blog | https://blogs.oracle.com/datamining/ |
This chapter explains how to install and configure Oracle Database to support Oracle Data Miner. This chapter contains these topics:

- Database Requirements for Oracle Data Miner
- Oracle Text and Oracle Data Miner
- XML DB and Oracle Data Miner
- Storage Configuration for Oracle Data Miner
- Installing Oracle Database 11.2 or 12.1

2.1 Database Requirements for Oracle Data Miner

Oracle Data Miner 4.1 requires Oracle Database Enterprise Edition or Personal Edition 11.2 or later. To verify that the database meets this requirement, you can query the `database_compatible_level` setting. The value should be no lower than 11.2.

```
SELECT VALUE FROM database_compatible_level;
```

**Note:** Some features of Oracle Data Miner 4.1 require Oracle Database 12.1.

Table 2–1 lists the Database installation and configuration requirements for Oracle Data Miner.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Requirement</th>
<th>Links to More Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database version</td>
<td>11.2.0.1 or higher.</td>
<td>Installing Oracle Database 11.2 or 12.1</td>
</tr>
<tr>
<td>Oracle Advanced Analytics</td>
<td>Required for Oracle Data Miner installation.</td>
<td>Oracle Data Miner and Oracle Advanced Analytics</td>
</tr>
<tr>
<td>Oracle Text</td>
<td>Required for Oracle Data Miner installation.</td>
<td>Oracle Text and Oracle Data Miner</td>
</tr>
<tr>
<td>XML DB</td>
<td>Required for Oracle Data Miner installation.</td>
<td>XML DB and Oracle Data Miner</td>
</tr>
<tr>
<td>Storage Configuration</td>
<td>Must be configured for either binary XML storage or object-relational storage, depending on the version of the database.</td>
<td>Storage Configuration for Oracle Data Miner</td>
</tr>
</tbody>
</table>
2.2 Oracle Text and Oracle Data Miner

Oracle Data Miner uses Oracle Text to support text mining. Oracle Text is included by default in Oracle Database Enterprise Edition and is required for installation of Oracle Data Miner.

Oracle Text Knowledge Base is required to support theme generation during text transformation. If you want to support this feature of Oracle Text and it is not already available in the Database, then you can obtain it with an installation of Oracle Database Examples. The Oracle Text Knowledge Base is available in English and French.

See Also: Oracle Database 12c:
- Oracle Text Application Developer’s Guide
- Oracle Text Reference for information about extending the supplied knowledge base
- Oracle Database Examples Installation Guide

Oracle Database 11g:
- Oracle Text Application Developer’s Guide
- Oracle Text Reference for information about extending the supplied knowledge base
- Oracle Database Examples Installation Guide

2.3 XML DB and Oracle Data Miner

Oracle Data Miner uses XML DB to store workflows in the Oracle Data Miner repository. XML DB is typically included in Oracle Database Enterprise Edition and is required for installation of Oracle Data Miner.

To determine if XML DB is present in the database, or to manually install XML DB, follow the instructions in Oracle XML DB Developer’s Guide.
Workflow documents in the Oracle Data Miner repository are of type XMLType, an abstract data type that provides these storage models:

- **Object-relational storage** – XMLType data is stored as a set of objects.
- **Binary XML storage** – XMLType data is stored in a post-parse, binary format specifically designed for XML data.

Oracle Data Miner uses object-relational storage in earlier versions of the Database and binary XML storage in later versions.

See Also: "Storage Configuration for Oracle Data Miner"

### 2.4 Storage Configuration for Oracle Data Miner

The storage format used by Oracle Data Miner depends on the version of the database:

- With Oracle Database 11.2.0.1 - 11.2.0.3, Oracle Data Miner uses object-relational storage.
  Approximately 200MB - 700MB XML DB object-relational storage is required.
- With Oracle Database 11.2.0.4 and higher, Oracle Data Miner uses binary XML storage.
  With binary storage, a tablespace allocation of 200MB can store approximately 1000 moderately complex workflows (about 35 nodes).

To use binary storage, ODMRSYS must be created in a tablespace managed with Automatic Segment Space Management (ASSM).

The following statement shows how to create an Oracle ASSM tablespace:

```
CREATE TABLESPACE DMUSER_AUTO DATAFILE 'DMUSER_AUTO.dat' size 20m
  autoextend on next 32m maxsize UNLIMITED extent management local
  AUTOALLOCATE SEGMENT SPACE MANAGEMENT AUTO;
```

See Also: Oracle Database Administrator’s Guide

### 2.5 Installing Oracle Database 11.2 or 12.1

To install Oracle Database, follow the installation instructions for your platform and all additional instructions specified in "Database Requirements for Oracle Data Miner".

Links to some Oracle Database Installation guides are listed as follows:

- **Linux**
  - Oracle Database 12.1: Oracle Database Installation Guide for Linux
  - Oracle Database 11.2: Oracle Database Installation Guide for Linux
- **Oracle Solaris**
  - Oracle Database 12.1: Oracle Database Installation Guide for Oracle Solaris
  - Oracle Database 11.2: Oracle Database Installation Guide for Oracle Solaris
- **Microsoft Windows**
  - Oracle Database 12.1: Oracle Database Installation Guide for Microsoft Windows
– Oracle Database 11.2: Oracle Database Installation Guide for Microsoft Windows

To install Oracle Database on other platforms, search the Oracle Help Center on http://docs.oracle.com for your platform-specific installation instructions.
This chapter explains how to install SQL Developer and the Oracle Data Miner repository. This chapter contains these topics:

- **Oracle Data Miner Installation Overview**
- **Downloading SQL Developer**
- **Installing the Repository Using a Script**
- **Installing the Sample Data Using a Script**
- **Installing the Repository Using the SQL Developer GUI**
- **Installing the JSON Parser and Data Guide**

### 3.1 Oracle Data Miner Installation Overview

*Oracle Data Miner installation* refers to installation of the Oracle Data Miner repository in Oracle Database. The repository serves as application manager and workflow storage manager for Oracle Data Miner.

To install Oracle Data Miner, perform these steps:

1. Install Oracle Database. If you already have a database, verify that it meets the requirements specified in Table 2–1.
2. Download SQL Developer.
3. Check *Oracle Data Miner Release Notes*.
4. Install the Oracle Data Miner repository. You can use SQL Developer or you can run scripts to perform the installation.

**See Also:**

- **Installing the Repository Using the SQL Developer GUI**
- **Installing the Repository Using a Script**
- "**Downloading SQL Developer**"
- "**Installing the Database to Support Oracle Data Miner**" for Oracle Database installation instructions
- "**About Data Miner Repository Installation**" in *Oracle Data Miner User’s Guide*
3.2 Downloading SQL Developer

Oracle SQL Developer is available for download free of charge on the Oracle Technology Network. To install SQL Developer, simply download and unzip it on your system. SQL Developer does not include an installation program.

To download SQL Developer:

1. Go to the Downloads tab of the Oracle SQL Developer home page:

2. Select the documentation links to view the release notes, a list of new features, and the SQL Developer Documentation Library.

   Note: The documents in the Documentation Library are available for online viewing or for download in PDF, Mobi (for Kindle), or ePub (for iBook) format. You can bookmark the Documentation Library page for ease of access:
   http://docs.oracle.com/cd/E55747_01/index.htm

3. Select the installation instructions for your platform and follow the instructions to download and start SQL Developer.

   Note: SQL Developer requires the Java Development Kit (JDK) version 1.7 or higher. If the JDK is not included in the SQL Developer software download bundle and you do not have it installed on your system, then you must download and install it separately.
   The SQL Developer installation instructions include a link to the JDK download page.

4. The first time you start SQL Developer, you must supply the path to the JDK.

3.3 Installing the Repository Using a Script

Before installing the repository, verify disk space availability. The Oracle Data Miner repository requires between 200 and 700 MB initially, depending on the tablespace settings.

Note: For Database 11.2.0.4 or later, the default tablespace for the repository must have auto specified for segment space management.

See "About the Oracle Data Miner Administration Scripts" before running the installation script.

To install the Oracle Data Miner repository:

1. Log in to the database as SYS.

2. Run the installodms script.

   installodmr.sql default_tablespace temp_tablespace
For example, if you have set the default search path as described in Section 5.1.1, then the following statement installs the repository with default tablespace USERS and temporary tablespace TEMP:

@installodmr USERS TEMP

When the database is remote and the repository is using XML DB object-relational storage (Oracle Database 11.2.0.1 - 11.2.0.3), the installation script takes approximately ten minutes to run. When the database is remote and the repository is using binary storage (Oracle Database 11.2.0.4 and higher), the installation completes in approximately three minutes.

Note: After installing the repository, you must enable Oracle Data Miner access for at least one database user.

See Also: Section 2.4, "Storage Configuration for Oracle Data Miner" for information about object-relational and binary storage for Oracle Data Miner.

"Granting or Dropping Access Rights to the Oracle Data Miner Repository" for more information related to enabling or disabling access to the repository.

### 3.4 Loading the Sample Data Using a Script

After you install the repository using a script, you can run a second script to load the sample data that is used in Oracle Data Miner tutorials. If you install the repository by using the SQL Developer GUI, then you can install the sample data by checking a check box.

The instDemoData script prepares demo data for an Oracle Data Miner user. The script grants access to the data described in Table 3–1. If the SH schema is not present in the database, then the script prepares only the demo data that does not depend on SH.

See Section 5.1, "About the Oracle Data Miner Administration Scripts" before running the installation scripts.

To install the Oracle Data Miner sample data for a user:

1. Log in to the database as SYS.
2. Verify that the SH schema is present in the database. If not, follow the instructions in Oracle Database Sample Schemas to install the sample schemas.
3. Run the instDemoData script:

   instDemoData.sql user

   For example, if you have set the default search path, then the following statement installs the sample data for the user dmuser1:

   @instDemoData dmuser1

To drop the sample data for a user:

1. Log in to the database as SYS.
2. Run the dropDemoData script:

   dropDemoData.sql user

   For example, if you have set the default search path, then the following statement drops the sample data for the user dmuser1:

   @dropDemoData dmuser1
3.4.1 About the Oracle Data Miner Sample Data

The Oracle Data Miner sample data includes the tables and views described in Table 3–1.

<table>
<thead>
<tr>
<th>Table or View</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MINING_DATA_BUILD_V</td>
<td>Views based on tables in the SH schema. This data is used in Oracle Data Mining sample programs, as described in:</td>
</tr>
<tr>
<td>MINING_DATA_TEST_V</td>
<td>Database 12.1: Oracle Data Mining Application Developer’s Guide</td>
</tr>
<tr>
<td>MINING_DATA_APPLU_V</td>
<td>Database 11.2: Oracle Data Mining Administrator’s Guide</td>
</tr>
<tr>
<td>MINING_DATA_TEXT_BUILD_V</td>
<td>Views based on tables in the SH schema. The views include an additional COMMENTS column that is used in text mining examples in OBEs and in Data Mining sample programs.</td>
</tr>
<tr>
<td>MINING_DATA_TEXT_TEST_V</td>
<td></td>
</tr>
<tr>
<td>MINING_DATA_TEXT_APPLU_V</td>
<td></td>
</tr>
<tr>
<td>ODMR_CARS_DATA</td>
<td>Data about US automobiles, for experimenting with the Graph node.</td>
</tr>
<tr>
<td>INSUR_CUST_LVT_SAMPLE</td>
<td>Data used by the Oracle By Example (OBE) tutorials for Data Mining.</td>
</tr>
<tr>
<td>ODMR_SALES_JSON_DATA</td>
<td>Sales data, for experimenting with the JSON query node. (Oracle Database 12.1.0.2 and higher)</td>
</tr>
</tbody>
</table>

3.5 Installing the Repository Using the SQL Developer GUI

To install the Oracle Data Miner repository from the GUI:

1. In the Connections tab of SQL Developer, select an administrative connection to the target database, or create a new one.
2. Create an Oracle Data Miner user:
   a. Drill on the administrative connection.
   b. Right click Other Users and choose Create User.
   c. Grant the CONNECT role to the user, and specify unlimited quota for the default tablespace.
3. In the Connections tab, create a connection for the Oracle Data Miner user.
4. From the View menu, select Data Miner, then Data Miner Connections.
5. The Data Miner Connections tab appears beneath the SQL Developer Connections tab.
6. Click the plus sign on the Data Miner tab to display the Select Connection dialog box. Choose the Data Miner user from the drop-down list.
7. The new connection appears on the Data Miner tab.
8. When you attempt to drill on the new connection, this dialog box is displayed:
9. Click Yes to install the repository.

10. After you provide the SYS password, information about the repository is displayed. For example:

11. Click OK.

12. The repository installation dialog box appears.

Select the Install Demo Data check box to install the sample data in the schema of the user connection. If the SH schema is present in the database, then all the sample data described in Table 3–1 is installed. If SH is not installed, then only ODMR_CARS_DATA, INSUR_CUST_LVT_SAMPLE, and ODMR_SALES_JSON_DATA (Database 12.1 only) are installed.

---

**Note:** The repository installation automatically grants the database privileges required by Oracle Data Miner.

The privileges required by Oracle Data Miner are different from the privileges required for using the Oracle Data Mining APIs directly. For details, see "Controlling Access to Mining Models and Data" in Oracle Data Mining Application Developer’s Guide.
13. When the installation is complete, click **Close**.

**See Also:** "Installing the Oracle Data Miner Repository Using the GUI" in *Oracle Data Miner User's Guide*

### 3.6 Installing the JSON Parser and Data Guide

JSON query processing is available in Oracle Database starting with version 12.1.0.2. If you have installed or migrated the repository using scripts, then you must complete a separate postinstallation step to enable JSON support in Oracle Data Miner. If you have installed or migrated the repository using the GUI, then this extra step is not required.

#### 3.6.1 About the Scripts for Installing JSON Support

Two scripts are available for installing the JSON parser and schema generator for Oracle Data Miner. The only difference between them is in their arguments: one takes the Oracle SID, the other takes the Oracle Service Name. You can choose the script that best suits your database installation. If you are using a pluggable database, the Oracle Service Name is required.

- **loadjsonschemagen.sql** uses the Oracle SID.
  
  ```sql
  loadjsonschemagen.sql sys_user sys_password host port SID jar_path
  ```

- **loadjsonschemagenWithSN.sql** uses the Oracle Service Name.
  
  ```sql
  loadjsonschemagenWithSN.sql sys_user sys_password host port service_name jar_path
  ```

For both scripts, specify the SYS user and password, the database host name and port number and the path to the jar files:

- `org.glassfish.javax.json.jar`
- `JSONSchemagenerator.jar`

#### 3.6.2 Requirements for Running the Scripts

- The `jar_path` argument must be a directory that is accessible to the current database system.
- The Oracle database command-line utilities `loadjava` and `dropjava` must be available on the system where the script is run.

#### 3.6.3 Example: Installing JSON Support in a Remote Database

1. Log in to the computer where SQL Developer is installed.
2. Navigate to the `scripts` directory:
   
   ```bash
   cd SQL_Developer_Home\sqldeveloper\dataminer\scripts
   ```
3. Start SQL*Plus as SYS:
   
   ```sql
   SQLPLUS sys / as sysdba
   ```
4. Run either the SID or Service Name version of the script:
   
   ```sql
   @loadjsonschemagen sys_user sys_password host port SID jar_path
   ```
   
   or
   
   ```sql
   @loadjsonschemagenWithSN sys_user sys_password host port service_name jar_path
   ```

   where the value of `jar_path` is:

   ```bash
   c:\SQL_Developer_home\sqldeveloper\extensions\oracle.dmt.dataminer\lib
   ```
3.6.4 Example: Installing JSON Support Locally on the Database Host

1. Copy the scripts `loadjsonschemagen.sql` and `validateODMRSYS.sql` from:
   \SQL_Developer_home\sqldeveloper\dataminer\scripts
to the database host computer staging directory (for example, /scratch).

2. Copy `org.glassfish.javax.json.jar` and `JSONSchemaGenerator.jar` from:
   \SQL_Developer_home\sqldeveloper\extensions\oracle.dmt.dataminer\lib
to the database host computer staging directory (for example, /scratch).

3. Go to the host computer staging directory (for example, /scratch) and start SQL*Plus as SYS:
   SQLPLUS sys / as sysdba

4. Run either the SID or Service Name version of the script.

   **See Also:** "About the Scripts for Installing JSON Support" for more information about how to run the scripts for installing the JSON parser.
Managing Oracle Data Miner Users

This chapter explains how to manage Oracle Data Miner user accounts. This chapter includes these topics:

- About User Objects and Repository Objects
- About Proxy Users for Oracle Data Miner
- Choosing an Access Model for Oracle Data Miner
- Granting or Dropping Access Rights to the Oracle Data Miner Repository
- Granting Access to Data

4.1 About User Objects and Repository Objects

An Oracle Data Miner installation consists of one repository and at least one user account. The user must have access to the repository and have the privileges required for data mining activities in the database, and appropriate access to data.

Oracle Data Miner stores information in the schema of the Oracle Data Miner user and in the repository schema, ODMRSYS. Mining models and data that support workflows are stored in the user’s schema. The metadata that defines the structure of projects and workflows is stored as XML documents in ODMRSYS.

See Also:

- "About the Oracle Data Miner Repository"
- "Oracle Data Miner Architecture for Big Data"

4.1.1 Objects in the Schema of the Oracle Data Miner User

Workflows create objects, tables and views. Oracle Data Miner stores these objects in the user’s schema:

- Objects, and they are visible to the user. These include mining models created by the Model node and tables created by the Create Table node.
- Tables and views, and they are not directly visible to the user. For example, the test results tables that are created during model build are internal. The user does not see the names of the tables, but the user can view the contents of the tables in the Test Results viewers.

4.1.2 About Oracle Data Miner Internal Tables

Internal tables in the user’s schema store information that supports workflows and data mining activities:
About Proxy Users for Oracle Data Miner

- The Data Mining engine creates tables with the DM$ prefix in the database. These tables store information about mining models.
- Oracle Data Miner creates tables with the ODMR$ prefix. These tables store information about workflows.

When you use the SQL Developer schema navigator to view the objects owned by an Oracle Data Miner user, the internal tables and views are included in the display. You can create a filter in SQL Developer to hide the internal tables and views. When you use Oracle Data Miner interfaces to view users’ schemas, the internal tables and views are automatically filtered out.

4.2 About Proxy Users for Oracle Data Miner

A connection is a SQL Developer object that specifies the login credentials for a specific user in a specific database. A Data Miner connection is a SQL Developer connection that includes the privileges required by a Data Miner user. Oracle Data Miner connections are listed in the Navigator on the Data Miner Connections tab.

A SQL Developer connection typically provides database access to a single user that is defined in that database. However, SQL Developer also provides support for proxy users that have their own login credentials but share the same target database user account. SQL Developer has several connection types that support the creation of proxy users.

Oracle Data Miner supports proxy authentication for Basic and TNS connection types. Figure 4–1 shows the SQL Developer Advanced Properties dialog box, which allows the creation of a proxy user for an existing Basic connection.
You can also use the SQL Developer LDAP service to create users that are functionally equivalent to proxy users. With LDAP, you create the individual (proxy) users and then associate them with an existing database user connection.

See Also:
- “Database Connections” in *Oracle SQL Developer User’s Guide*
- “Connections with Proxy Authentication” in *Oracle SQL Developer User’s Guide*
- Information on LDAP connections in *Oracle SQL Developer User’s Guide*
- "About Proxy Authentication” in *Oracle Database Security Guide*

4.3 Choosing an Access Model for Oracle Data Miner

You can choose to limit Oracle Data Miner access to a single database user, or you can enable multiple database users with access. Either way, you can create proxy users so that groups of people can share one Data Miner user account in the database.
- **Single User Access** — In this scenario, there is one user schema. Either one person can use Oracle Data Miner or a group of people with proxy accounts can use Oracle Data Miner. Proxy users have access to the same models and database objects. All users can create, modify, and drop database objects, and all users see the results of other users’ work.

- **Multiple User Access** — In this scenario, there are multiple user schemas. A schema can support an individual user, or it can support a workgroup of proxy users, or you can set up some combination of individual and shared access. You can use proxy authentication for all users, even for unshared users.

The access model that you choose depends on the number of users that you need to support and whether the users need to collaborate in a shared environment or work independently in a private environment.

### 4.3.1 About Shared Access

Shared user environments facilitate collaboration, but users must take care to avoid interfering with the work of other users.

When several proxy users share access to a single database account, Oracle Data Miner uses locking mechanisms to coordinate access to workflows. Workflows are locked while they are executing or waiting to execute, or when they are being edited.

The name space for workflows is a project. When several users work in the same project, they should take care to name their workflows in a way that distinguishes them from the workflows of other users. For example, users could agree to prefix their workflow names with their initials.

The name namespace for database objects, such as mining models and tables, is unique within the shared schema. Oracle Data Miner follows naming conventions for database objects to ensure uniqueness. If a user overrides the system-generated name for a table that is referenced in another workflow, then a warning is generated.

See Also: Oracle Data Miner User’s Guide for more information about naming and renaming workflows.

#### 4.3.1.1 About the Document in Use Condition

If an Oracle Data Miner client disconnects from the network (for example, if a cable is disconnected or a laptop goes into deep sleep), locks on the workflows are not released. The disconnected session is still locked and running in the database. If another user tries to edit the workflow, a Document in Use message is generated.

You can attempt to reclaim the lock by clicking the lock on the tool bar. If you are unable to reclaim the lock, then you must stop the database session that is holding the locks. Refer to "Terminating sessions" in Oracle Database Administrator’s Guide for instructions.

### 4.3.2 About Single User Access

In the absence of proxies, a user functions autonomously within its own schema. The security mechanisms of Oracle Database prevent users from modifying objects that belong to another user’s schema. Single user access ensures private workspaces but does not promote collaboration.
4.4 Granting or Dropping Access Rights to the Oracle Data Miner Repository

You can grant access rights to the Oracle Data Miner repository using the GUI or by running a script. You can revoke access rights by running a script.

- Granting Access Rights Using the GUI
- Granting Access Rights Using a Script
- Dropping Access Rights Using a Script

4.4.1 Granting Access Rights Using the GUI

When you install the Oracle Data Miner repository using the GUI, access rights to the repository are automatically granted to your user ID. If you logged in as a proxy or LDAP user, Oracle Data Miner automatically grants the access rights to the target user.

When you select a connection for the first time to a database that already has the repository installed, you are prompted to confirm that you want to grant access rights and, optionally, install the sample data.

See Also:  ■ Table 3–1, "Oracle Data Miner Sample Data"
- "Installing the Repository Using the SQL Developer GUI"

4.4.2 Granting Access Rights Using a Script

You can grant access rights to the repository by executing the usergrants script and specifying a user name. The repository must be already installed before you run the script.

usergrants.sql user_access
For example, the following statement grants Oracle Data Miner access to the user dmuser1:

@usergrants dmuser1
The user name that you specify must be a target user. Any proxy or LDAP users that authenticate based on this target user automatically acquire the permissions of the target user.

4.4.3 Dropping Access Rights Using a Script

You can drop access rights to the Oracle Data Miner repository by executing the dropusergrants script.

dropusergrants.sql user_access
For example, the following statement drops the access rights that were granted to dmuser1.

@dropusergrants dmuser1
As with the usergrants script, the user name that you specify must be a target user. Any proxy or LDAP users that authenticate based on this target user automatically acquire the permissions of the target user. When you drop the access rights for the target user, all proxy and LDAP users that are based on that target user automatically lose access to the repository.

See Also:  "Granting Access Rights Using a Script"
4.5 Granting Access to Data

Users must have read access (`SELECT` permission) to data that they use for building mining models or for scoring.

You must grant `SELECT` permission directly to a target user. Do not grant the permission indirectly to a user role. The `SELECT` permission must be granted directly so that Oracle Data Miner can create views on behalf of the user. If Oracle Data Miner cannot create views, then the user may not be able to access the data.
This chapter explains how to use scripts to manage the Oracle Data Miner repository. This chapter contains these topics:

- About the Oracle Data Miner Administration Scripts
- Determining the Status and Version of the Repository
- Backing Up and Restoring the Repository
- Migrating the Repository
- Dropping the Repository

5.1 About the Oracle Data Miner Administration Scripts

Oracle Data Miner includes a set of SQL scripts for installing and managing the repository. The scripts are installed with SQL Developer in the following directory:

`\SQL_Developer_Home\sqldeveloper\dataminer\scripts`

You can run the scripts in SQL*Plus or in SQL Developer Worksheet. All the Oracle Data Miner scripts must be run as SYS.

**Note:** Many of the Oracle Data Miner scripts are integrated with SQL Developer, enabling access to some administrative functions through the Data GUI.

5.1.1 Setting the Path to the Oracle Data Miner Scripts

To set the default search path for scripts:

- **SQL*Plus:** Start SQL*Plus from the `scripts` directory.

  `\SQL_Developer_Home\sqldeveloper\dataminer\scripts`

- **SQL Developer Worksheet:** Set the default search path to the `scripts` directory in the Worksheet properties.

  Also in SQL Developer Worksheet properties, you should change the maximum number of rows to print in a script to 500000.

5.2 Determining the Status and Version of the Repository

The version of the repository must be compatible with the version of the Oracle Data Miner client. If the client and server versions are not compatible, then the client cannot connect to the server.
The following query returns the repository version and status:

```sql
set echo on;
-- value of VERSION and REPOSITORY_STATUS
SELECT property_name, property_str_value
FROM ODMRSYS.ODMR$REPOSITORY_PROPERTIES
WHERE property_name IN ('VERSION', 'REPOSITORY_STATUS', 'WF_VERSION');
```

<table>
<thead>
<tr>
<th>PROPERTY_NAME</th>
<th>PROPERTY_STR_VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>REPOSITORY_STATUS</td>
<td>LOADED</td>
</tr>
<tr>
<td>VERSION</td>
<td>12.1.0.2.3</td>
</tr>
<tr>
<td>WF_VERSION</td>
<td>12.1.0.2.3</td>
</tr>
</tbody>
</table>

The Oracle Data Miner repository has two values for status: NOT_LOADED and LOADED. The status NOT_LOADED, usually indicates that the Repository is being upgraded to support a new version of SQL Developer or a new patch release. When the upgrade completes, then the status is LOADED.

### 5.3 Backing Up and Restoring the Repository

Before upgrading the Oracle Data Miner repository or performing a database upgrade, you should perform a full backup of Oracle Data Miner, including ODMRSYS and the Oracle Data Miner user schemas. Oracle Data Miner also provides scripts for backing up the workflow metadata in ODMRSYS without including the user schemas.

#### See Also:
- "About User Objects and Repository Objects"
- "Oracle Data Miner Architecture for Big Data"

### 5.3.1 Full Backup and Restore

For databases 11.2.0.4 and higher, you can perform a full backup and restore of the Oracle Data Miner repository and user schemas independently of a full database backup. In databases 11.2.0.1 - 11.2.0.3, a full Oracle Data Miner backup is only possible as part of a full database backup.

#### 5.3.1.1 Full Backup and Restore in Database 11.2.0.1 to 11.2.0.3

In Oracle Database 11.2.0.1, 11.2.0.2, and 11.2.0.3, the XML storage in the Oracle Data Miner repository is object-relational. In these databases, there is no mechanism for backing up and restoring the Oracle Data Miner repository and user schemas independently of a full backup and restore of the database.

To backup and restore the database, you can use Oracle Recovery Manager (Oracle RMAN). When restored from backup, the database will be exactly as it was before the backup. A partial restore is not supported.

#### See Also: Oracle Database Backup and Recovery User’s Guide for information about Oracle RMAN

#### 5.3.1.2 Full Backup and Restore in Database 11.2.0.4 and Higher

In Oracle Database 11.2.0.4 and higher, the XML storage in the Oracle Data Miner repository is binary. In these databases, you can use Oracle Data Pump to backup and restore ODMRSYS and the user schemas independently of a full backup and restore of the database. With Oracle Data Pump, you can backup and restore individual schemas.

Alternatively you can backup and restore Oracle Data Miner with Oracle RMAN.
5.3.2 Workflow Only Backup

Oracle Data Miner provides a script for backing up the workflow metadata in the repository without including the objects in the users’ schemas that are generated by the workflows. This simplified backup strategy safeguards the workflow specifications and enables you to restore a workflow if a user accidentally deletes it. After the workflow is restored, the user would have to re-run it to ensure that all the supporting objects are present in the user’s schema.

The `createxmlworkflowsbackup2` script backs up all the workflows in `ODMRSYS` to a table called `ODMR$WORKFLOWS_BACKUP` in a separate backup account. Before you run the script, ensure that the backup schema exists and is available.

Syntax:

```
createxmlworkflowsbackup2.sql backup_account
```

Parameter:

- `backup_account` is the name of the schema of the backup table, `ODMR$WORKFLOWS_BACKUP`.

Example:

This example backs up the workflows in a backup account called `BACKUPACCT`:

```
set serveroutput on
@createxmlworkflowsbackup2 BACKUPACCT
```

Each time you run `createxmlworkflowsbackup2`, a full set of workflows is added to the backup table. The backup script maintains up to 30 distinct backups within the backup table. Older backups are automatically deleted. For example, if the backup was run each day, then a user has up to 30 days to request a restore of a workflow.

In the backup script, the `DEFINE_MAX_VERSIONS` specifies the number of backups that are preserved in the backup table. If you want to preserve more than 30 backups, then in the backup script `createxmlworkflowsbackup2`, change the value of `DEFINE_MAX_VERSIONS` to the desired number.

See Also: "Dropping the Repository" for more information about the `dropRepositoryAndUserObjects` script.

5.3.3 Workflow Only Restore

To restore the workflows from the backup table created by `createxmlworkflowsbackup2`, run the `restorexmlworkflowfrombackup2` script.

```
restorexmlworkflowfrombackup2.sql restore workflows from the backup table to the Oracle Data Miner repository. Use it as follows:
```

Syntax:
Parameters:

`option` is an optional parameter that can have one of the following values:

- **ADD_ONLY** — Restore workflows that do not already exist in the repository, creating missing projects if necessary. (Default)
- **DROP_AND_ADD** — Drop all existing workflows and projects in the repository, then restore all workflows from backup, creating missing projects if necessary.
- **OVERRIDE_ONLY** — Only restore workflows that already exist in the repository.
- **OVERRIDE_AND_ADD** — Applies both the OVERRIDE_ONLY and ADD_ONLY options.

`backup_account` is optional unless `workflow_definition` is specified, in which case it is required. If no backup account is specified, then workflows are restored from the backup table in the repository. If the backup tables does not exist, then an exception is raised.

`workflow_definition` is an optional parameter that identifies a table or view that specifies which workflows to restore from backup. The table or view must contain these four columns: USER_NAME, PROJECT_NAME, WORKFLOW_NAME, and VERSION. Each row in the table identifies a workflow to restore. If the VERSION number is null, then the latest version number is used for the restore. When no workflow definition is provided, then the latest backup version is the default.

Example:

This example drops all the workflows in the repository and restores the workflows from the backup table in BACKUPACCT.

```
set serverput on
@restorexmlworkflowfrombackup2.sql ADD_ONLY BACKUPACCT
```

See Also: "Workflow Only Backup" for more information about restoring workflows from backup tables.

### 5.3.4 Workflow Only Restore Examples

This topic provides examples on selective workflow restore and full workflow restore.

**Example 5-1  Selective Workflow Restore**

Let’s assume the user SCOTT had accidentally deleted all his workflows last week. You can use the ADD_ONLY option to restore his workflows. You will have to query the backup table to determine which version of backups contain his missing workflows. If the version is 12, then the following script example, run as SYS will reload only those workflows.

```
@restorexmlworkflowfrombackup2.sql ADD_ONLY BACKUPACCT BACKUPACCT.WORKFLOW_V
```

The WORKFLOW_V view, shown as follows, selects all the workflows present for the user SCOTT from a specified version backup number.

```
CREATE VIEW BACKUPACCT.WORKFLOW_V AS
    SELECT user_name, project_name, workflow_name, version
    FROM backupacct.odmr$workflows_backup
    WHERE user_name='SCOTT' AND version = 12;
```

**Example 5-2  Full Workflow Restore**

Let’s assume that there was some critical repository failure that requires a full reload of all workflows from the latest backup. You can use the DROP_AND_ADD option to
insure that all the old workflows are dropped and all the workflows on the backup are reloaded. In this case, the backup table is located in another account separate from the ODMRSYS account. The latest backup version will be used for the recovery, so no workflow definition parameter is required.

@restorexmlworkflowfrombackup2.sql DROP_AND_ADD BACKUPACCT

5.4 Migrating the Repository

Starting with SQL Developer 4.0, Oracle Data Miner migration scripts are available for specific upgrade paths. The scripts, described in Table 5–1, upgrade the Oracle Data Miner repository from an earlier version to the current version. All workflows are preserved during the upgrade.

See Also: "About Data Miner Repository Migration" in Oracle Data Miner User's Guide

Table 5–1  Oracle Data Miner Upgrade Scripts

<table>
<thead>
<tr>
<th>SQL Developer Version</th>
<th>Database Version</th>
<th>Script</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>any version</td>
<td>11.2.0.1 - 11.2.0.3</td>
<td>migrateodmr.sql</td>
<td>Upgrades ODMRSYS. See Section 5.4.1 for details.</td>
</tr>
<tr>
<td>3.2.2 or earlier</td>
<td>11.2.0.4 and higher, where the default ODMRSYS tablespace is not ASSM-based</td>
<td>upgradeRepoWithNewTableSpace.sql</td>
<td>ASSM tablespace required as input parameter. This will be used to migrate the workflow data from XML object storage to XML binary storage. See Section 5.4.2 for details.</td>
</tr>
<tr>
<td>3.2.2 or earlier</td>
<td>11.2.0.4 and higher, where the default ODMRSYS tablespace is ASSM-based.</td>
<td>upgradeReposFromORtoBinary.sql</td>
<td>Workflow data will be migrated from XML object storage to XML binary storage. Section 5.4.3 for details.</td>
</tr>
<tr>
<td>4.0 and higher</td>
<td>11.2.0.4 and higher</td>
<td>migratebinaryodmr.sql</td>
<td>Relevant for future releases, when XML conversion from object storage to binary storage will no longer needed. Section 5.4.4 for details.</td>
</tr>
<tr>
<td>4.0 and higher</td>
<td>11.2.0.3 or lower originally, and then the database was upgraded to 11.2.0.4 of higher</td>
<td>upgradeReposWithNewTableSpace.sql or upgradereposFromORtoBinary.sql</td>
<td>Since the database can be upgraded independently of the Oracle Data Miner repository, this has to be taken into account. The choice of scripts will depend on whether ODMRSYS tablespace is ASSM or not. See Section 5.4.2 or Section 5.4.3 for details.</td>
</tr>
</tbody>
</table>

Note: The Oracle Data Miner version number is the same as the SQL Developer version number as listed in Table 5–1.
5.4.1 Upgrade ODMRSYS

The `migrateodmr` script upgrades ODMRSYS to the latest version that is supported in the database.

If any sessions that have the ODMRUSER role are currently running, then the session object locks block the upgrade. You can use the `session_disconnect` parameter to disconnect any active sessions, thus enabling the upgrade process to proceed.

**Syntax:**

```
migrateodmr.sql session_disconnect
```

**Parameters:**

- `session_disconnect` can have one of the following values:
  - `R` — Report active sessions. Do not disconnect them.
  - `D` — Disconnect active sessions. Do not report them.
  - `DR` or `RD` — Disconnect and report active sessions.

**Example:**

This example upgrades ODMRSYS, disconnecting and reporting any active ODMRUSER sessions.

```
@migrateodmr DR
```

5.4.2 Upgrade New ODMRSYS Tablespace From Object-Relational to Binary

The `upgradeRepoWithNewTableSpace` script upgrades the specified tablespace from object-relational to binary XML storage and migrates the workflow data in ODMRSYS to the new tablespace. The new tablespace must be an Oracle Automatic Segment Space Management (Oracle ASSM) tablespace.

If any sessions that have the ODMRUSER role are currently running, then the session object locks block the upgrade. You can use the `session_disconnect` parameter to disconnect any active sessions, thus enabling the upgrade process to proceed.

**Syntax:**

```
upgradeRepoWithNewTableSpace.sql ASSMtablespace session_disconnect
```

**Parameters:**

- `ASSMtablespace` is the name of an ASSM tablespace.
- `session_disconnect` can have one of the following values:
  - `R` — Report active sessions. Do not disconnect them.
  - `D` — Disconnect active sessions. Do not report them.
  - `DR` or `RD` — Disconnect and report active sessions.

**Example:**

This example migrates object-relational XML data in ODMRSYS to the new ASSM tablespace, `my_ASSM_space`, that uses binary XML storage. If any ODMRUSER sessions are active, they are disconnected and reported.

```
@upgradeRepoWithNewTableSpace my_ASSM_space DR
```

**See Also:** Oracle Database Administrator’s Guide
5.4.3 Upgrade ODMRSYS From Object-Relational to Binary

The `upgradeRepoFromORtoBinary` script migrates the ODMRSYS workflow data from object-relational XML storage to binary XML storage. The upgraded ODMRSYS tablespace is an Oracle Automatic Segment Space Management (ASSM) tablespace.

If any sessions that have the ODMRUSER role are currently running, then the session object locks block the upgrade. You can use the `session_disconnect` parameter to disconnect any active sessions, thus enabling the upgrade process to proceed.

**Syntax:**

```
@upgradeRepoFromORtoBinary.sql session_disconnect
```

**Parameters:**

`session_disconnect` can have one of the following values:

- R — Report active sessions. Do not disconnect them.
- D — Disconnect active sessions. Do not report them.
- DR or RD — Disconnect and report active sessions.

**Example:**

This example upgrades ODMRSYS from object-relational XML storage to binary XML storage. The upgraded tablespace is ASSM-based. If any ODMRUSER sessions are active, they are disconnected and reported.

```
@upgradeRepoFromORtoBinary DR
```

**See Also:** Oracle Automatic Storage Management Administrator’s Guide

5.4.4 Upgrade Binary Workflow Data in ODMRSYS

The `migratebinaryodmr` script upgrades the workflow data in an ODMRSYS schema that uses binary XML storage. The ODMRSYS tablespace is an Oracle Automatic Segment Space Management (ASSM) tablespace.

If any sessions that have the ODMRUSER role are currently running, then the session object locks block the upgrade. You can use the `session_disconnect` parameter to disconnect any active sessions, thus enabling the upgrade process to proceed.

**Syntax:**

```
@migratebinaryodmr.sql session_disconnect
```

**Parameters:**

`session_disconnect` can have one of the following values:

- R — Report active sessions. Do not disconnect them.
- D — Disconnect active sessions. Do not report them.
- DR or RD — Disconnect and report active sessions.

**Example:**

This example upgrades the binary XML workflow data in ODMRSYS, disconnecting and reporting any active ODMRUSER sessions.

```
@migratebinaryodmr DR
```
5.5 Dropping the Repository

The `dropRepositoryAndUserObjects` script drops the Oracle Data Miner repository and related objects in the users' schemas.

If any sessions that have the `ODMRUSER` role are currently running, then the session object locks block the upgrade. You can use the `session_disconnect` parameter to disconnect any active sessions, thus enabling the upgrade process to proceed.

**Syntax:**

`dropRepositoryAndUserObjects.sql session_disconnect`

**Parameters:**

- `session_disconnect` can have one of the following values:
  - R — Report active sessions. Do not disconnect them.
  - D — Disconnect active sessions. Do not report them.
  - DR or RD — Disconnect and report active sessions.

**Example:**

This example drops the `ODMRSYS` schema and related objects in the Oracle Data Miner users' schemas, disconnecting and reporting any active `ODMRUSER` sessions.

```
@dropRepositoryAndUserObjects DR
```

**See Also:**

- "About User Objects and Repository Objects"
- "Oracle Data Miner Architecture for Big Data"
- "Workflow Only Backup"
This chapter provides information to help you optimize your system to support Oracle Data Miner. This chapter contains these topics:

- Oracle Data Miner Resource Management Overview
- Allocating Resources to Oracle Data Miner User Sessions
- Managing Model Builds
- Managing Workflow Execution
- Managing Parallel Execution
- Summary of Oracle Data Miner Repository Properties for System Management

### 6.1 Oracle Data Miner Resource Management Overview

Using features of Oracle Database and options provided in the Oracle Data Miner repository, you can effectively manage system resources for Oracle Data Miner as follows:

- To manage Oracle Data Miner sessions, develop an appropriate resource plan using Oracle Database Resource Manager.
- To manage workflow execution, change the Oracle Data Miner default job class in Oracle Scheduler to a job class with an appropriate resource plan.
- To manage the model build process, change the Oracle Data Miner default maximum number of concurrent model builds.
- To manage parallel execution, change the Oracle Data Miner default parallel query setting to prevent users from specifying parallel execution for individual nodes and workflows.

### 6.2 Allocating Resources to Oracle Data Miner User Sessions

In a database where multiple applications run concurrently and compete for system resources, you can use Oracle Database Resource Manager to distribute the workloads and optimize overall performance. For example, you could balance the demands of ETL, OLAP, data mining, and reporting workloads running simultaneously in the database.

With Database Resource Manager, you can allocate CPU time, configure parallel execution, limit the number of sessions, and control other aspects of system behavior that would otherwise be controlled by the operating system. Using Database Resource
Manager, you can create resource plans that allocate system resources for groups of sessions based on session attributes.

Oracle Data Miner workflows can potentially make extensive demands on system resources, especially when transformations and large data sets are involved. In a database that must accommodate the demands of multiple applications, you can create a resource plan to limit the impact of Oracle Data Miner sessions on other applications and prevent other applications from compromising the performance of Oracle Data Miner.

Example 6–1 illustrates the creation of a simple resource plan for Oracle Data Miner. The resource plan, called SIMPLE_RESOURCE_PLAN, creates two consumer groups: DATA_MINER_GROUP and OTHER_GROUPS. The plan allocates 50% of CPU resource to DATA_MINER_GROUP and the rest to OTHER_GROUPS. The DATA_MINER_GROUP is mapped to the DMUSER account; other users are mapped to the OTHER_GROUPS group.

**Example 6–1 Simple Resource Plan for Oracle Data Miner Sessions**

```
CONNECT sys as sysdba;
Enter password: password
-- creating a pending area is the first step in defining
-- consumer groups and resource plans
EXEC dbms_resource_manager.create_pending_area();
-- delete old plan (optional)
EXEC dbms_resource_manager.delete_plan_cascade(  
  plan => 'SIMPLE_RESOURCE_PLAN');
-- create a custom consumer group for data miner workload
EXEC dbms_resource_manager.create_consumer_group(  
  consumer_group => 'DATA_MINER_GROUP',
  comment => 'Sessions for data miner operations');
-- map DMUSER account to the consumer group
EXEC dbms_resource_manager.set_consumer_group_mapping(  
  attribute => dbms_resource_manager.oracle_user,
  value => 'DMUSER',
  consumer_group => 'DATA_MINER_GROUP');
-- create a custom resource plan
EXEC dbms_resource_manager.create_plan(  
  plan => 'SIMPLE_RESOURCE_PLAN',
  comment => 'Resource plan for database operations');
-- specifies how much CPU and parallelism
-- should be allocated to the consumer group
EXEC dbms_resource_manager.create_plan_directive(  
  plan => 'SIMPLE_RESOURCE_PLAN',
  group_or_subplan => 'DATA_MINER_GROUP',
  comment => 'Percentage of CPU for DATA_MINER_GROUP',
  mgmt_p1 => 50,
  utilization_limit => 55,
  parallel_degree_limit_p1 => 8,
  parallel_server_limit => 4);
-- specifies how much CPU should be allocated to the required OTHER_Groups
EXEC dbms_resource_manager.create_plan_directive(  
  plan => 'SIMPLE_RESOURCE_PLAN',
  group_or_subplan => 'OTHER_GROUPS',
  comment => 'Percentage of CPU for OTHER_GROUPS',
  mgmt_p1 => 50);
-- persist plan to the database
EXEC dbms_resource_manager.submit_pending_area();
-- Now that the resource plan is defined, enable it by setting
-- the resource_manager_plan parameter with the resource plan name
ALTER SYSTEM SET resource_manager_plan = 'SIMPLE_RESOURCE_PLAN';
```
Managing Model Builds

The process of building mining models can consume significant system resources. To control the impact of model builds on overall system resources, you can raise or lower the value of the `MAX_NUM_THREADS` repository property.

`MAX_NUM_THREADS` specifies the maximum number of mining model builds that can execute concurrently across all workflows in an Oracle Data Miner session. `MAX_NUM_THREADS` has no effect on model builds that are triggered individually and do not belong to a workflow.

For example, if one workflow is attempting to build 25 models while another workflow is attempting to build 15 models and `MAX_NUM_THREADS` is 10, then 10 model build operations occur simultaneously and the 30 remaining model builds are queued. The delayed build processes complete with a warning message that explains the reason for the delay. Two additional properties, `THREAD_WAIT_TIME` and `MAX_THREAD_WAIT` control the queuing of model builds. See Table 6–1 for details.

**Example 6–2** shows how to increase the maximum number of concurrent model builds from 10 (the default) to 15.

```
Example 6–2 Changing the Number of Concurrent Model Builds

set echo on;
-- value of MAX_NUM_THREADS before update
SELECT property_name, property_num_value
FROM ODMRSYS.ODMR$REPOSITORY_PROPERTIES
WHERE property_name = 'MAX_NUM_THREADS';
-- update MAX_NUM_THREADS
UPDATE ODMRSYS.ODMR$REPOSITORY_PROPERTIES
SET property_num_value = 15
WHERE property_name = 'MAX_NUM_THREADS';
-- commit change
COMMIT;
-- value of MAX_NUM_THREADS after update
SELECT property_name, property_num_value
FROM ODMRSYS.ODMR$REPOSITORY_PROPERTIES
WHERE property_name = 'MAX_NUM_THREADS';

This script produces the following log:

-- value of MAX_NUM_THREADS before update
SELECT property_name, property_num_value
FROM ODMRSYS.ODMR$REPOSITORY_PROPERTIES
WHERE property_name = 'MAX_NUM_THREADS';
PROPERTY_NAME PROPERTY_NUM_VALUE
--------------------------- ----------------------
MAX_NUM_THREADS 10
-- update MAX_NUM_THREADS
UPDATE ODMRSYS.ODMR$REPOSITORY_PROPERTIES
SET property_num_value = 15
WHERE property_name = 'MAX_NUM_THREADS';
1 rows updated
-- commit change
commit
```

See Also: "Managing Resources With Oracle Resource Manager" in Oracle Database Administrator’s Guide.
committed
-- value of MAX_NUM_THREADS after update
SELECT property_name, property_num_value
FROM ODMRSYS.ODMR$REPOSITORY_PROPERTIES
WHERE property_name = 'MAX_NUM_THREADS';
PROPERTY_NAME PROPERTY_NUM_VALUE
------------------------------- ----------------------
MAX_NUM_THREADS 15

6.4 Managing Workflow Execution

Oracle Data Miner submits workflows to Oracle Database Scheduler for execution as Scheduler jobs. Oracle Scheduler supports a variety of features that control how system resources are allocated. You can configure Oracle Scheduler to effectively manage a large pool of run requests.

Oracle Data Miner uses the default Scheduler job class, DEFAULT_JOB_CLASS as its own default. In a resource plan, jobs that run as DEFAULT_JOB_CLASS are not assigned to any consumer group; access to system resources is not restricted for jobs that have the default class. You can change the job class to a class that is based on a consumer group by setting the Oracle Data Miner repository property WORKFLOW_JOB_CLASS.

Example 6–3 shows you could create a MINING_CLASS job class based on a consumer group HEAVY_LOAD_RESOURCE_GROUP, which was previously created to allocate high CPU for heavy workload jobs. When you update WORKFLOW_JOB_CLASS, the workflow will run with access to system resources that are restricted to this consumer group. The resource plan for the assigned HEAVY_LOAD_RESOURCE_GROUP group must be active when the workflow is run. You can set up Scheduler windows to activate specific resource plans at specific time periods.

Example 6–3 Changing the Scheduler Job Class for Oracle Data Miner Workflows

connect sys as sysdba;
Enter password: password
EXEC DBMS_SCHEDULER.CREATE_JOB_CLASS(
    job_class_name => 'MINING_CLASS',
    resource_consumer_group => 'HEAVY_LOAD_RESOURCE_GROUP');
GRANT EXECUTE ON MINING_CLASS to DMUSER;
-- update WORKFLOW_JOB_CLASS
UPDATE ODMRSYS.ODMR$REPOSITORY_PROPERTIES
SET property_str_value = 'MINING_CLASS'
WHERE property_name = 'WORKFLOW_JOB_CLASS';
-- commit change
commit;
-- verify value of WORKFLOW_JOB_CLASS after update
SELECT property_name, property_str_value
FROM ODMRSYS.ODMR$REPOSITORY_PROPERTIES
WHERE property_name = 'WORKFLOW_JOB_CLASS';

See Also:

- Oracle Database Administrator’s Guide for information about administering Oracle Scheduler
- Oracle Database PL/SQL Packages and Types Reference for information about DBMS_SCHEDULER
6.5 Managing Parallel Execution

Oracle Data Miner workflows and viewers, and most Data Mining algorithms, can take advantage of parallel execution when it is enabled in the database. Parameters in INIT.ORA control the behavior of parallel execution. By default parallelism is disabled (PARALLEL_DEGREE_POLICY=MANUAL).

Parallel execution can be system-determined (PARALLEL_DEGREE_POLICY=AUTO), or it can be set to Degree of Parallelism (PARALLEL_DEGREE_POLICY=AUTO). When parallel execution is system-determined, the database dynamically determines Degree of Parallelism values for all SQL statements.

The parallel feature of Oracle Database is designed to use maximum resources assuming the operation will finish faster if you use more resources. In a multiuser environment, increasing the use of parallelism can rapidly deplete system resources, reducing resources for other users to execute parallel statements concurrently.

Oracle Data Miner workflows support a parallel query feature, which allows users to manually enable parallel execution for specific nodes or entire workflows. You can disable this feature by setting the Oracle Data Miner repository property PARALLEL_QUERY_ON_ALLOWED to FALSE. By default this property is set to TRUE. Example 6–4 shows how to disable the parallel query feature of Oracle Data Miner.

Example 6–4 Disabling the Parallel Query Feature for Oracle Data Miner Workflows and Nodes

connect sys as sysdba;
Enter password: password
-- value of PARALLEL_QUERY_ON_ALLOWED before update
SELECT property_name, property_str_value
FROM ODMRSYS.ODMR$REPOSITORY_PROPERTIES
WHERE property_name = 'PARALLEL_QUERY_ON_ALLOWED';
-- update PARALLEL_QUERY_ON_ALLOWED
UPDATE ODMRSYS.ODMR$REPOSITORY_PROPERTIES
SET property_str_value = 'FALSE'
WHERE property_name = 'PARALLEL_QUERY_ON_ALLOWED';
-- commit change
COMMIT;
-- verify value of PARALLEL_QUERY_ON_ALLOWED after update
SELECT property_name, property_str_value
FROM ODMRSYS.ODMR$REPOSITORY_PROPERTIES
WHERE property_name = 'PARALLEL_QUERY_ON_ALLOWED';

The Oracle Data Miner model build process can generate many parallel sessions if your database configuration allows for it. To limit the impact on the overall system, you should implement an appropriate resource plan and, if necessary, prevent users from setting parallel query within their Data Miner sessions.

See Also: Oracle Database VLDB and Partitioning Guide for an introduction to parallel execution

6.6 Summary of Oracle Data Miner Repository Properties for System Management

Table 6–1 provides a summary of the system management properties available in the Oracle Data Miner repository.
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PARALLEL_QUERY_ON_ALLOWED</td>
<td>Boolean</td>
<td>Indicates whether users can specify parallel query for nodes or workflows. Values are TRUE or FALSE. Default is TRUE.</td>
</tr>
<tr>
<td>MAX_NUM_THREADS</td>
<td>Integer</td>
<td>Maximum number of concurrent model builds. Default is 10.</td>
</tr>
<tr>
<td>THREAD_WAIT_TIME</td>
<td>Integer</td>
<td>When MAX_NUM_THREADS is reached, any outstanding model build processes are queued until the parallel model build count is less than MAX_NUM_THREADS. The THREAD_WAIT_TIME setting determines how long to wait before checking the parallel model build count. The default wait time is 5 seconds.</td>
</tr>
<tr>
<td>MAX_THREAD_WAIT</td>
<td>Integer</td>
<td>The timeout interval in seconds for a model build process that has been queued. When a timeout occurs, the build process exits with an error message stating that the process lock wait timeout interval has been exceeded. When the value is NULL, no timeout occurs. Default is NULL</td>
</tr>
<tr>
<td>WORKFLOW_JOB_CLASS</td>
<td>Varchar</td>
<td>Oracle Scheduler job class for workflows. Default is DEFAULT_JOB_CLASS.</td>
</tr>
</tbody>
</table>
7 Generating and Deploying SQL Scripts

SQL Script Generation is a feature that is available in Oracle Data Miner 4.1. This chapter provides an overview and a use case to illustrate this feature. Topics include:

- Overview of the SQL Script Generation Feature
- Overview of the SQL Script Generation Use Case
- Generating SQL Script Files from the Workflow
- Scheduling Workflow Script Files
  - Scheduling SQL Scripts using SQL Developer
  - Scheduling SQL Scripts using Oracle Enterprise Manager
- Deploying SQL Scripts on the Target Database

7.1 Overview of the SQL Script Generation Feature

Oracle Data Miner 4.1 provides the SQL script generation feature. By using this feature, you can generate SQL scripts for one or all nodes in a workflow. You can then integrate the SQL scripts into another application. In this way, Oracle Data Miner provides the scope to integrate data mining with another end user application. This feature is explained in detail supported by a use case.

See Also: "Overview of the SQL Script Generation Use Case" for more information.

7.1.1 Audience

The primary target audience for this content are:

- Application Developers
- Data Analysts
- Database Administrators
- IT Management Professionals

7.2 Overview of the SQL Script Generation Use Case

The SQL script generation feature is explained with the help of a use case, that uses a sample workflow codegen_workflow and a demo database table INSUR_CUST_LTV_SAMPLE. The use case demonstrates how to:

- Import, run, and deploy the workflow codegen_workflow
Generate SQL script from the workflow codegen_workflow

Schedule SQL scripts to run on the database using:
- Oracle SQL Developer
- Oracle Enterprise Manager

Deploy the generated SQL scripts on a Target or Production Database

See Also: "Premise of the Use Case"
- "Prerequisites"
- "Location of Demo Workflow Files"
- "About the Sample Workflow"

7.2.1 Premise of the Use Case

The premise on which the use case is built are:

- The Data Analysts define the workflow for model building and scoring.
- The Data Analysts use the new script generation feature to hand over a set of SQL scripts to the Application Developer for deployment.
- The Application Developers deploy the scripts to the target or production database, where they can schedule the scripts to run periodically. This allows the model to be built with fresh customer data every time the scripts are run.
- Demo Database: The use case uses a demo database table INSUR_CUST_LTV_SAMPLE, which can be installed to a users account.
- Predefined workflow: The use case explains the procedure with the help of the predefined workflow codegen_workflow.

See Also: About the Sample Workflow
- Prerequisites
- Location of Demo Workflow Files
- Loading the Sample Data Using a Script

7.2.1.1 Location of Demo Workflow Files

The workflow file codegen_workflow.xml which contains the predefined workflow is available in the SQL Developer installation location at:
<sqldeveloper>/dataminer/demos/workflows.

7.2.2 About the Sample Workflow

The sample workflow in this use case codegen_workflow, comprises two distinct processes contained within a single lineage: Modeling (top) and Scoring (bottom) as shown in Figure 7–1. Both the processes use the demo data INSUR_CUST_LTV_SAMPLE as the input data source.
Modelling: The modeling process builds a Classification Support Vector Machine (SVM) model. It predicts whether the customer will buy insurance or not. The model coefficients are persisted to a database table for viewing. This table may provide a basis for application integration.

Scoring: The scoring process makes predictions for the customer data using the Support Vector Machine (SVM) model created by the modeling lineage. The prediction result is persisted to a database view for viewing. The view provides the following:

- Predictions of the current input data. For example, if the input table is refreshed with new data, this view will automatically capture the predictions of the new data.
- Basis for application integration.

See: ■ "Premise of the Use Case"
■ "Prerequisites"
■ "Location of Demo Workflow Files"

### 7.2.3 Prerequisites

Before deploying a workflow, ensure the following:

1. Install Oracle Data Miner 4.1 on your system.
2. Create an Oracle Data Miner user account.

Here is an example of a sample statement to create a Data Mining user account. This statement must be issued by a privileged user.

```sql
grant create session, create table, create view, create mining model, create procedure, unlimited tablespace to <username>;
```

3. Load the database table.
4. Import and run the workflow.
7.2.3.1 Importing and Running the Workflow

To import the predefined workflow_codegen.xml:

1. In SQL Developer 4.1, go to the Data Miner tab and expand the connection.
2. Right-click the connection and click New Project.
3. Right-click the project that you just created and select Import Workflow.
4. In the Import Workflow dialog box, browse to the location where you have downloaded and saved the sample workflow file codegen_workflow.xml. Select the codegen_workflow.xml and click OK. The imported workflow codegen_workflow is now displayed in the Oracle Data Miner UI, as shown in Figure 7–2.

See Also: • "Installing Oracle Data Miner" for the procedure to install Oracle Data Miner 4.1.
• Oracle By Example Tutorials for the procedures to create a Data Miner user account, and a SQL Developer connection for the Data Miner user account, at: http://www.oracle.com/webfolder/technetwork/tutorials/ob/12c/r1/dm/ODM12c-SetUp.html
• "Importing and Running the Workflow" for the procedure to import and run the workflow.
• "Location of Demo Workflow Files" to import the demo workflow codegen_workflow.
• "Loading the Sample Data Using a Script" for more information about how to load a database table.

Figure 7–2 The codegen_workflow after import

5. Right-click the INSUR_CUST_LTV_SAMPLE_BUILD node, and click Force Run.
6. In the Force Run submenu, select the option Selected Node and Children. This runs all the nodes in the codegen_workflow. Once the workflow is run successfully, all the nodes in the workflow are depicted with the green check mark, as shown in Figure 7–3.
This completes the task of importing and running the workflow.

See Also:  
- “Generating SQL Script Files from the Workflow”
- “Scheduling Workflow Script Files”
- “Deploying SQL Scripts on the Target Database”
- ”Location of Demo Workflow Files” to import the demo workflow file `codegen_workflow.xml`.

7.3 Generating SQL Script Files from the Workflow

You must run the workflow before generating the SQL script from it. To generate the SQL script files from a workflow:

1. Right-click any node and select Deploy. For this use case, right-click the INSUR_CUST_LTV_SAMPLE BUILD node, and select Selected node and connected node option under Deploy.

   Note: For this use case, the predefined workflow `codegen_workflow` is used.

The script deployment options are:

- **Selected Node and dependent nodes**: This option generates the SQL script for the selected node and all its parent nodes. For example, as shown in Figure 7–3, if the Apply Node is selected, then a script will be generated for these nodes:
  - INSUR_CUST_LTV_SAMPLE BUILD
  - Class Build
  - INSUR_CUST_LTV_SAMPLE APPLY
  - Apply

- **Selected node, dependent nodes and children nodes**: This option generates script for the selected node and all its parents and children nodes. For example, in Figure 7–3, if the Apply Node is selected, then a script will be generated for these nodes:
  - INSUR_CUST_LTV_SAMPLE BUILD
  - Class Build
Selected nodes and connected nodes: This option generates scripts for the selected node and all the nodes that are connected to the selected node. In this example, as shown in Figure 7–3, if the Apply Node is selected, then a script will be generated for all the nodes that are connected to the Apply node in the workflow.

Note: To generate a script for the entire workflow, you can select all the nodes in the workflow (by clicking all the nodes while pressing Ctrl key simultaneously) and select any of the above three deployment options.

2. After selecting a Deploy options, the Generate SQL Script - Step 1 of 2 wizard opens, as shown in Figure 7–4.

Figure 7–4 Generate SQL Script - Step 1 of 2 Wizard

3. In the Target Database Version field, select the database version. Ensure that the generated script is compatible with the version of the database that you select here. This is the database where the SQL script will run.

4. Click Next. The Generate SQL Script - Step 2 of 2 window opens, as shown in Figure 7–5.
5. In the Step 2 of the Generate Script Wizard, provide the following information:
   - **Script Directory**: This is the name of the directory where the generated scripts are stored.
   - **Base Directory**: Click **Browse** to navigate to another location and create the directory for the scripts.
   - **Directory Path**: Displays the path of the script directory.

6. Click **Finish**. This triggers the script generation process. Once the scripts are generated successfully, the following message is displayed, as shown in Figure 7–6. Click **OK**.

   **Figure 7–6 Deploy Code Dialog Box**

You can check the list of generated scripts in the script directory that you defined in step 5.
See Also:  ■ "About the Sample Workflow"
        ■ "List of Generated SQL Script Files"

### 7.3.1 List of Generated SQL Script Files

Table 7–1 lists the SQL script files that are generated from the `codegen_workflow`, along with their descriptions:

<table>
<thead>
<tr>
<th>Script File Type</th>
<th>Script File Name</th>
<th>Examples of Script Files Generated from the <code>codegen_workflow</code></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master Script</td>
<td><code>&lt;workflow name&gt;_Run.sql</code></td>
<td><code>codegen_workflow_Run.sql</code></td>
<td>Invokes all the required node level scripts in the correct order. It performs the following tasks:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Validates compatibility of the version of the script file with the Data Miner Repository version.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Creates a workflow master table that contains entries for all the underlying objects created by the workflow script.</td>
</tr>
<tr>
<td>Cleanup Script</td>
<td><code>&lt;workflow name&gt;_Drop.sql</code></td>
<td><code>codegen_workflow_Drop.sql</code></td>
<td>Drops all objects created by the master script. It drops the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Hidden objects, such as tables generated for Explore Data nodes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Public objects such as model names created by the Build nodes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>■ Tables created by the Create Table node.</td>
</tr>
<tr>
<td>Workflow Image</td>
<td><code>&lt;workflow name&gt;.png</code></td>
<td><code>codegen_workflow.png</code></td>
<td>This is an image of the workflow at the time of script generation.</td>
</tr>
<tr>
<td>Node Script</td>
<td><code>&lt;node name&gt;.sql</code></td>
<td><code>Apply.sql</code></td>
<td>Performs node specific operations, such as Model creation in Build nodes. One node script is generated for each node that participates in the script generation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>Class Build.sql</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>Coefficients.sql</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>INSUR_CUST_LTV_SAMPLE APPLY.sql</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>INSUR_CUST_LTV_SAMPLE BUILD.sql</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>MODEL_COEFFICIENTS.sql</code></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>SCORED_CUSTOMERS.sql</code></td>
<td></td>
</tr>
</tbody>
</table>

### 7.3.2 Variable Definitions in Script Files

SQL scripts generated for the nodes have variable definitions that provide object names for the public objects created by the scripts. The master script invokes all the underlying node level scripts in proper order. Therefore, all variable definitions must be defined in the master script.

The following variables are supported:

■ Variables that allow you to change the name of the objects that are input to the node level scripts, such as tables or views, and models. By default, these names are the original table or view names, and model names.
Variables that allow you to change the name of the Control table. By default, the name of the Control table is the workflow name.

Variables that indicate if named objects should be deleted first before they are generated by the script.

### 7.3.3 Control Tables

When the master script `<workflow name>_Run.sql` is run, the Control Table is created first by using the name specified in the control table name variable. The Control Table performs the following:

- Registers generated objects, such as views, models, text specifications and so on
- Allows input objects to be looked up by the logical nodes in the workflow, and registers their output objects
- Determines the objects that need to be dropped by the cleanup script
- Provides internal name of objects that are not readily accessible through the workflows. For example, users can find the model test result tables by viewing the Control Table.
- By using different control file names along with different output variable names, you can use the generated script to concurrently generate and manage different results. This may be useful if the input data sources continue different sets of data that you want to mine independently. In this use case, the application would be responsible for saving the name of the Control Table so that it can be utilized when rerunning or dropping the generated results.

**See Also:**
- "Querying the Control Table"
- "Structure of the Control Table"

### 7.3.3.1 Structure of the Control Table

The structure of the Control Table is as follows:

```sql
CREATE TABLE "&WORKFLOW_OUTPUT"
(
    NODE_ID VARCHAR2(30) NOT NULL,
    NODE_NAME VARCHAR2(30) NOT NULL,
    NODE_TYPE VARCHAR2(30) NOT NULL,
    MODEL_ID VARCHAR2(30),
    MODEL_NAME VARCHAR2(65),
    MODEL_TYPE VARCHAR2(35),
    OUTPUT_NAME VARCHAR2(30) NOT NULL,
    OUTPUT_TYPE VARCHAR2(30) NOT NULL,
    ADDITIONAL_INFO VARCHAR2(65),
    CREATION_TIME TIMESTAMP(6) NOT NULL,
    COMMENTS VARCHAR2(4000 CHAR)
)
```

See Also:
- "Querying the Control Table"
- "Structure of the Control Table"
**See Also:** Table 7–2 for more information about the columns in the Control Table, their description and examples.

### 7.3.3.2 Columns of the Control Table

Table 7–2 lists the columns in the Control Table along with their description and examples.

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>NODE_ID</td>
<td>This is the ID of the node that constitutes a part of the workflow. It uniquely identifies the node.</td>
<td>10001, 10002</td>
</tr>
<tr>
<td>NODE_NAME</td>
<td>This is the name of the node that constitutes a part of the workflow.</td>
<td>Class Build, MINING_DATA_BUILD_V</td>
</tr>
<tr>
<td>NODE_TYPE</td>
<td>This is the category of node.</td>
<td>Data Source node, Class Build node and so on.</td>
</tr>
<tr>
<td>MODEL_ID</td>
<td>This is the ID of the workflow model. It uniquely identifies each model referenced within a workflow.</td>
<td>10101, 10102</td>
</tr>
<tr>
<td>MODEL_NAME</td>
<td>This is the name of the model.</td>
<td>CLAS_GLM_1_6</td>
</tr>
<tr>
<td>MODEL_TYPE</td>
<td>Model type is the algorithm type used by the model.</td>
<td>Generalized Linear Model, Support Vector Machines and so on</td>
</tr>
<tr>
<td>OUTPUT_NAME</td>
<td>This is the name of the output. These are internally generated names unless the names are under the control of the user.</td>
<td>Table/View Name, Model Name, Text object names such as: ODMR$15_37_21_839599RMAFRXI - table name &quot;DMUSER&quot;.&quot;CLAS_GLM_1_6&quot; - fully qualified model name</td>
</tr>
<tr>
<td>OUTPUT_TYPE</td>
<td>It qualifies the type of output object.</td>
<td>Table, view, model</td>
</tr>
<tr>
<td>ADDITIONAL_INFO</td>
<td>This is the information that qualifies the purpose of the object about the script execution.</td>
<td>Target class for test lift result</td>
</tr>
<tr>
<td>CREATION_TIME</td>
<td>This is the time of object creation.</td>
<td>11-DEC-12 03.37.25.935193000 PM (format determined by locale)</td>
</tr>
</tbody>
</table>
7.4 Scheduling Workflow Script Files

All the generated SQL script files must be deployed to the target or production database where they are accessible by the database instance. The SQL script files must be stored together in the same directory.

This section shows how to use SQL Developer and Oracle Enterprise Manager to schedule the master script to run.

- **Prerequisites for Scheduling Workflow Script Files**
- **Scheduling SQL Scripts using SQL Developer**
- **Scheduling SQL Scripts using Oracle Enterprise Manager**

### 7.4.1 Prerequisites for Scheduling Workflow Script Files

The prerequisites to schedule the SQL script files are:

- **Oracle Database**: An instance of Oracle Database is required to schedule the generated SQL script files.
- **SQL script files**: All the generated SQL script files should be deployed to the target or production database host, where they are accessible by the database instance. All the scripts files should be stored together in the same directory.
- **Oracle Data Miner Repository**: The Data Miner Repository is required to run the scripts because some node scripts use the services provided by the Repository at runtime. Some examples of services provided by the Repository are statistic calculation for Explorer node, text processing for Build Text node and so on.
- **Oracle Data Miner user account**: The user account is required to run the script files because it has the necessary grants to the services provided by the Repository.
- **Complete directory path in the master script file**: Add the complete directory path to each node script invocation in the master script. This is required so that the individual node script files can be called by the master script during runtime.

**See:** "Adding Complete Directory Path"

### 7.4.1.1 Adding Complete Directory Path

To add the complete directory path in the master script:
1. Open the master script `codegen_workflow_Run.sql`, and locate the following lines:

```sql
-- Workflow run
@@"INSUR_CUST_LTV_SAMPLE BUILD.sql";
@@"INSUR_CUST_LTV_SAMPLE APPLY.sql";
@@"Class Build.sql"; @@"Coefficients.sql";
@@"MODEL_COEFFICIENTS.sql";
@@"Apply.sql";
@@"SCORED_CUSTOMERS.sql";
```

2. Edit the master script file to add the complete directory path, where the scripts will be stored in the target or production database host computer. In this example, it is assumed that the script files will be deployed to `home/workspace` directory. For example:

```sql
-- Workflow run
@@"/home/workspace/INSUR_CUST_LTV_SAMPLE BUILD.sql";
@@"/home/workspace/INSUR_CUST_LTV_SAMPLE APPLY.sql";
@@"/home/workspace/Class Build.sql";
@@"/home/workspace/Coefficients.sql";
@@"/home/workspace/MODEL_COEFFICIENTS.sql";
@@"/home/workspace/Apply.sql";
@@"/home/workspace/SCORED_CUSTOMERS.sql";
```

3. Save and close the master script file.

### 7.4.1.2 Creating Credentials for Database Host and Database

A credential is an Oracle Scheduler object that has a user name and password pair stored in a dedicated database object. You must create two credentials for:

- **Host credential**: A SQLPlus script job uses a host credential to authenticate itself with a database instance or the operating system so that the SQLPlus executable can run.

- **Connection credential**: This credential contains a database credential, which connects SQLPlus to the database before running the script.

To create the credentials:

1. In the **Connections** tab, expand the connection in which your user account is created.

2. Expand **Scheduler** under that connection.

3. Under **Scheduler**, right-click **Credentials** and click **New Credentials**. The Create Credentials dialog box opens.

4. First, create the host credential to log in to the host on which the job is running. Provide the following information:
   - **Name**
   - **Select Enabled**.
c. Description
d. User Name
e. Password

5. Click Apply.

6. Next, create the connection credential for the Database connection. Repeat the same procedure as described in step 1 through step 5.

This completes the task of creating credentials for the database host and connection.

7.4.2 Scheduling SQL Scripts using SQL Developer

Oracle SQL Developer provides the graphical user interface to define Scheduler Jobs. Scheduling SQL scripts using SQL Developer involves the following:

1. Creating Credentials for Database Host and Database
2. Defining Scheduler Job using Job Wizard

7.4.2.1 Defining Scheduler Job using Job Wizard

To define job schedules:

1. In the SQL Developer Connections tab, expand the connection in which your user account is created.

2. Expand Scheduler under that connection.


4. In the Create Job Wizard - Step 1 of 6 dialog box, define the Job Details with the following information:
   a. Job Name:
   b. Select Enabled.
   c. Description:
   d. Job Class:
   e. Type of Job: Select Script.
   f. Script Type: Select SQLPlus.
   g. When to Execute Job: Select Repeat Interval.
   h. In the Repeat Interval dialog box, set the repeat interval, start date, time and click OK.
   i. Click Next.

5. In the Create Job Wizard - Step 2 of 6 dialog box, define the following:
   a. Select the option Local from the drop-down list.
   b. Select Credential: Select the host credential that you created in Creating Credentials for Database Host and Database from the drop-down list.
   c. Connect Credential Name: Select the connection credential that you created in Creating Credentials for Database Host and Database from the drop-down list.
   d. Click Next.
6. In the Create Job Wizard - Step 4 of 6 dialog box, you can set up email notification based on the job status.
   a. In the Select Events section, select the job events for which you want to send email notifications.
   b. In the Recipients field, enter the email address. For each message, you can specify recipient email addresses and the sender (optional).
   c. Click Next.

7. In the Create Job Wizard - Step 5 of 6 dialog box, click Next. For this use case, this step is skipped.

8. In the Create Job Wizard - Step 6 of 6 dialog box, click Finish. This completes the creation of the job schedule.

After creating the job, you can monitor it in SQL Developer.

7.4.3 Scheduling SQL Scripts using Oracle Enterprise Manager

Oracle Enterprise Manager allows Database Administrators to define jobs. The job definition defines the master script invocation as a script file using a full file path. You can decide whether the job should be run on a schedule or on demand. You can also monitor the running of the job in the application.

To schedule jobs in Oracle Enterprise Manager:

1. Log in to the Oracle Enterprise Manager using your Oracle Database account.
2. In the Job Activity section, click Jobs. The Job Creation page opens.
3. In the Create Job drop-down list, select SQL Script and click Go. This opens the Create Job page where you can define the new job.
4. In the General tab, enter the following details:
   a. Name
   b. Description
   c. Target Database: Provide the target database where the job will run.
5. In the Parameters tab, provide the full path name of the cleanup script and master script in the SQL Script section.
6. In the Credentials tab, provide the credentials for the following:
   ■ Database Host Credential
   ■ Database Credentials
7. In the Schedule tab, define the schedule of the job.
8. In the Access tab, you can set up email notifications based on the job status.
9. Click Submit to create the job.

See Also: "Creating Credentials for Database Host and Database"

7.5 Deploying SQL Scripts on the Target Database

Deploying the SQL scripts involves running the master script file from the base directory, as follows:

> @"C: <base directory>\<workflow name>_Run.sql"
For example, run the master script `codegen_workflow_Run.sql` in SQLPlus from the base directory, as follows:

```sql
> C:\code_gen\codegen workflow\codegen_workflow_Run.sql
```

If you must run the master script file subsequently, run the cleanup script `codegen_workflow_Drop.sql` first to delete previously generated objects, and then run the master script, as follows:

```sql
> C:\code_gen\codegen workflow\codegen_workflow_Drop.sql
> C:\code_gen\codegen workflow\codegen_workflow_Run.sql
```

### 7.5.1 Querying the Control Table

After running the SQL scripts, you can query the Control Table to examine the generated objects. To query the Control Table, run this command in SQLPlus:

```sql
> select * from <workflow_name>
```

For example, query the Control Table for `codegen_workflow` to examine the generated objects, as follows:

```sql
> select * from "codegen_workflow"
```

![Figure 7–7 An Output Table - Result of the Query](image)

In this example, the Create Table node `MODEL_COEFFICIENTS`, produced an output table `MODEL_COEFFICIENTS` that persisted the coefficient data extracted from the generated SVM model.

**See Also:**

- "Control Tables"
- "Structure of the Control Table"
This chapter explains how to manage workflows using the PL/SQL APIs. This chapter includes these topics:

- About PL/SQL APIs
- PL/SQL APIs
- Repository Views
- PL/SQL APIs Use Cases

### 8.1 About PL/SQL APIs

Oracle Data Miner 4.1 ships with a set of repository PL/SQL APIs that enable applications to manage Oracle Data Miner projects and workflow directly. The project PL/SQL APIs is in the ODMR_PROJECT package, and the workflow PL/SQL APIs is in the ODMR_WORKFLOW package. Both the packages are defined in the ODMRSYS schema in the Oracle Data Miner repository.

The PL/SQL APIs enable you to:

- Manage Data Miner projects and workflows
- Schedule workflows
- Run workflows
- Query project and workflow information
- Monitor workflow execution status
- Query generated results

### 8.2 PL/SQL APIs

You can use the PL/SQL APIs to perform the following tasks:

- PROJECT_CREATE: To create a project
- PROJECT_RENAME: To rename a project
- PROJECT_DELETE: To delete one or more projects
- WF_RUN: To run a workflow
- WF_STOP: To stop a running workflow
- WF_RENAME: To rename a workflow
- WF_DELETE: To delete one or more workflows
8.2.1 PROJECT_CREATE

The function PROJECT_CREATE creates a project using the project name that you provide. The function returns a project ID.

If the project already exists, the function raises an exception.

Function:

FUNCTION PROJECT_CREATE(p_project_name IN VARCHAR2,
p_comment IN VARCHAR2 DEFAULT NULL) RETURN NUMBER

Table 8–1 lists the parameters that are used in the PROJECT_CREATE function.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_project_name</td>
<td>Assign a name to the project that is created.</td>
</tr>
<tr>
<td>p_comment</td>
<td>Specify any comment that is to be applied to the project.</td>
</tr>
</tbody>
</table>

8.2.2 PROJECT_RENAME

The PROJECT_RENAME procedure renames an existing project. If a project with the new name already exists, then the procedure raises an exception.

Procedure:

PROCEDURE PROJECT_RENAME(p_project_id IN NUMBER, p_project_name IN VARCHAR2)

Table 8–2 lists the parameters that are used in the PROJECT_RENAME procedure.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_project_id</td>
<td>Specify the project ID of the project to rename.</td>
</tr>
<tr>
<td>p_project_name</td>
<td>Specify the new name for the project.</td>
</tr>
</tbody>
</table>

8.2.3 PROJECT_DELETE

The procedure PROJECT_DELETE enables you to delete one or more projects along with the workflows contained in it. If any workflow is running or is opened by Oracle Data Miner, then the procedure raises an exception.

Procedure to delete a project:

PROCEDURE PROJECT_DELETE(p_project_id IN NUMBER)

Procedure to delete multiple projects:

PROCEDURE PROJECT_DELETE(p_project_ids IN ODMR_OBJECT_IDS)

Table 8–3 lists the parameters that are used in the PROJECT_DELETE procedure.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_project_id</td>
<td>Specify the project ID of the project to delete.</td>
</tr>
</tbody>
</table>
8.2.4 WF_RUN

The function `WF_RUN` that runs a workflow contains signatures that accepts names, project IDs, workflow and specific nodes to run. The project ID, workflow ID, and node IDs can be queried using the `ODMR_USER_WORKFLOW_NODES` view.

You can execute the `WF_RUN` function using different parameter combinations:

- **WF_RUN with Project Name, Workflow Name, and Node Name**
- **WF_RUN with Project ID, Workflow ID and Node IDs**
- **WF_RUN with Project Name, Workflow Name Node Name and Time Interval**
- **WF_RUN with Project ID, Workflow ID, Node ID and Time Interval**

The `RERUN_WORKFLOW` RUN mode runs all nodes in a workflow regardless of how these nodes are connected. If a workflow contains two or more separate lineage of nodes, all lineages will be run, but the order of lineage execution is not deterministic. That is, the user cannot set the order for the lineage to run.

Table 8–4 lists the parameters that are used in the `WF_RUN` function.

### Table 8–4  List of Parameters in the WF_RUN function

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>P_PROJECT_NAME</code></td>
<td>Specify the project name that the workflow was created in.</td>
</tr>
<tr>
<td><code>P_PROJECT_ID</code></td>
<td>Specify the project ID that the workflow was created in.</td>
</tr>
<tr>
<td><code>P_WORKFLOW_NAME</code></td>
<td>Specify the workflow name to run.</td>
</tr>
<tr>
<td><code>P_WORKFLOW_ID</code></td>
<td>Specify the workflow ID to run.</td>
</tr>
<tr>
<td><code>P_NODE_NAMES</code></td>
<td>Specify the node names in the workflow to run.</td>
</tr>
<tr>
<td><code>P_NODE_IDS</code></td>
<td>Specify the node IDs in the workflow to run.</td>
</tr>
<tr>
<td><code>P_RUN_MODE</code></td>
<td>■ VALIDATE_ONLY: Validates parent nodes of the specified nodes.</td>
</tr>
<tr>
<td></td>
<td>■ RUN_NODE_ONLY: Runs the specified nodes. If the nodes have already run, they will not run again. If parent nodes have not run, then they will be run, otherwise they will be ignored.</td>
</tr>
<tr>
<td></td>
<td>■ RERUN_NODE_ONLY: Resets the status of the specified nodes to READY state. Then these nodes are run again.</td>
</tr>
<tr>
<td></td>
<td>■ RERUN_NODE_CHILDREN: Resets the status of the specified nodes and their children nodes to READY state. Then these nodes are run again.</td>
</tr>
<tr>
<td></td>
<td>■ RERUN_NODE_PARENTS: Resets the status of the specified nodes and their parent nodes to READY state. Then these nodes are run again.</td>
</tr>
<tr>
<td></td>
<td>■ RERUN_WORKFLOW: Resets the status of all nodes to READY state. Then the nodes are run (complete workflow run). Note: <code>p_node_names</code> is ignored.</td>
</tr>
</tbody>
</table>
### PL/SQL APIs

8.2.4.1 *WF_RUN* with Project Name, Workflow Name, and Node Name

The *WF_RUN* function with the project name, workflow name and node name parameters:

```sql
FUNCTION WF_RUN(P_PROJECT_NAME IN VARCHAR2,
    P_WORKFLOW_NAME IN VARCHAR2,
    P_NODE_NAMES IN ODMR_OBJECT_NAMES,
    P_RUN_MODE IN VARCHAR2 DEFAULT 'RUN_NODE_ONLY',
    P_MAX_NUM_THREADS IN NUMBER DEFAULT NULL,
    P_SCHEDULE IN VARCHAR2 DEFAULT NULL,
    P_JOB_CLASS IN VARCHAR2 DEFAULT NULL)
RETURN VARCHAR2
```

8.2.4.2 *WF_RUN* with Project ID, Workflow ID and Node IDs

The *WF_RUN* function with the project ID, workflow ID and node ID parameters:

```sql
FUNCTION WF_RUN(P_PROJECT_ID IN NUMBER,
    P_WORKFLOW_ID IN NUMBER,
    P_NODE_IDS IN ODMR_OBJECT_IDS,
    P_RUN_MODE IN VARCHAR2 DEFAULT 'RUN_NODE_ONLY',
    P_MAX_NUM_THREADS IN NUMBER DEFAULT NULL,
    P_SCHEDULE IN VARCHAR2 DEFAULT NULL,
    P_JOB_CLASS IN VARCHAR2 DEFAULT NULL)
RETURN VARCHAR
```

8.2.4.3 *WF_RUN* with Project Name, Workflow Name Node Name and Time Interval

The *WF_RUN* function with the name parameters and start date and end date:

```sql
FUNCTION WF_RUN(P_PROJECT_NAME IN VARCHAR2,
    P_WORKFLOW_NAME IN VARCHAR2,
    P_NODE_NAMES IN ODMR_OBJECT_NAMES,
    P_MAX_NUM_THREADS Specify the maximum number of parallel model builds across all workflows. Specify NULL for system determined. Use this parameter only if your system has plenty of resources, otherwise set this value to NULL to use the default value.

P_SCHEDULE Specify existing schedule object defined in the Scheduler. If no value is specified for P_SCHEDULE, then the workflow is scheduled to run as soon as possible.

P_START_DATE Specify the date and time on which this workflow is scheduled to start for the first time. If P_START_DATE and P_REPEAT_INTERVAL are set to NULL, then the workflow is scheduled to run as soon as possible.

P_REPEAT_INTERVAL Specify how often the workflow repeats. You can specify the repeat interval by using the calendar or PL/SQL expressions. The expression specified is evaluated to determine the next time the workflow should run. If P_REPEAT_INTERVAL is not specified, then the workflow runs only once at the specified start date.

P_END_DATE Specify the date and time after which the workflow expires and is no longer run. If no value for P_END_DATE is specified, then the job repeats indefinitely.

P_JOB_CLASS Specify existing job class to run the workflow. If no value for P_JOB_CLASS is specified, then the default job class is used.
```

---

**Table 8–4 (Cont.) List of Parameters in the *WF_RUN* function**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_MAX_NUM_THREADS</td>
<td>Specify the maximum number of parallel model builds across all workflows. Specify NULL for system determined. Use this parameter only if your system has plenty of resources, otherwise set this value to NULL to use the default value.</td>
</tr>
<tr>
<td>P_SCHEDULE</td>
<td>Specify existing schedule object defined in the Scheduler. If no value is specified for P_SCHEDULE, then the workflow is scheduled to run as soon as possible.</td>
</tr>
<tr>
<td>P_START_DATE</td>
<td>Specify the date and time on which this workflow is scheduled to start for the first time. If P_START_DATE and P_REPEAT_INTERVAL are set to NULL, then the workflow is scheduled to run as soon as possible.</td>
</tr>
<tr>
<td>P_REPEAT_INTERVAL</td>
<td>Specify how often the workflow repeats. You can specify the repeat interval by using the calendar or PL/SQL expressions. The expression specified is evaluated to determine the next time the workflow should run. If P_REPEAT_INTERVAL is not specified, then the workflow runs only once at the specified start date.</td>
</tr>
<tr>
<td>P_END_DATE</td>
<td>Specify the date and time after which the workflow expires and is no longer run. If no value for P_END_DATE is specified, then the job repeats indefinitely.</td>
</tr>
<tr>
<td>P_JOB_CLASS</td>
<td>Specify existing job class to run the workflow. If no value for P_JOB_CLASS is specified, then the default job class is used.</td>
</tr>
</tbody>
</table>
PL/SQL APIs

8.2.4.4 WF_RUN with Project ID, Workflow ID, Node ID and Time Interval

The WF_RUN function with the IDs and start date and end date parameters:

```
FUNCTION WF_RUN(P_PROJECT_ID IN NUMBER,
    P_WORKFLOW_ID IN NUMBER,
    P_NODE_IDS IN ODMR_OBJECT_IDS,
    P_RUN_MODE IN VARCHAR2 DEFAULT 'RUN_NODE_ONLY',
    P_MAX_NUM_THREADS IN NUMBER DEFAULT NULL,
    P_START_DATE IN TIMESTAMP WITH TIME ZONE DEFAULT NULL,
    P_REPEAT_INTERVAL IN VARCHAR2 DEFAULT NULL,
    P_END_DATE IN TIMESTAMP WITH TIME ZONE DEFAULT NULL,
    P_JOB_CLASS IN VARCHAR2 DEFAULT NULL)
RETURN VARCHAR2
```

8.2.5 WF_STOP

The procedure WF_STOP enables you to stop or cancel a workflow that is scheduled to run. If the workflow is not already running or scheduled, then the procedure raises an exception. The procedure is:

```
PROCEDURE WF_STOP(p_workflowId IN NUMBER)
```

Table 8–5 lists the parameters that are used in the WF_STOP procedure.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_workflow_id</td>
<td>Specify the workflow ID of the workflow to cancel.</td>
</tr>
</tbody>
</table>

8.2.6 WF_RENAME

The procedure WF_RENAME renames an existing workflow. The procedure raises an exception under the following conditions:

- If a workflow with the new name already exists
- If the workflow is either already running or opened by Oracle Data Miner

The procedure to rename the workflow:

```
PROCEDURE WF_RENAME(p_workflowId IN NUMBER,
    p_workflow_name IN VARCHAR2,
    p_mode IN CHAR DEFAULT 'R')
```

Table 8–6 lists the parameters that are used in the WF_RENAME procedure.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_workflow_id</td>
<td>Specify the workflow ID to rename.</td>
</tr>
<tr>
<td>p_workflow_name</td>
<td>Specify the new workflow name.</td>
</tr>
<tr>
<td>p_mode</td>
<td>This parameter is for internal use only.</td>
</tr>
</tbody>
</table>
8.2.7 WF_DELETE

The procedure `WF_DELETE` deletes a workflow along with all the generated objects such as tables, views, models, test results, and so on. If the workflow is either already running or opened by the Oracle Data Miner, then it raises an exception.

Procedure:

```sql
PROCEDURE WF_DELETE(p_workflowId IN NUMBER)
```

Table 8–7 lists the parameters that are used in the `WF_DELETE` procedure.

**Table 8–7 List of Parameters for WF_DELETE Procedure**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_workflow_id</td>
<td>Specify the ID of the workflow that is to be deleted.</td>
</tr>
</tbody>
</table>

8.2.8 WF_IMPORT

The `WF_IMPORT` function imports a workflow (exported by the Oracle Data Miner) to the specified project. Since the workflow is backward compatible, you can import an older version of a workflow to a newer Oracle Data Miner Repository. During import, the function detects if the workflow has any object name conflicts with the existing workflows in the repository. The `p_force` parameter determines whether to terminate the import or not.

Exceptions are raised under the following conditions:

- If the project does not exist
- If the workflow metadata is invalid or incompatible with the current repository
- If a workflow with the same name already exists

Function:

```sql
FUNCTION WF_IMPORT(p_project_id IN NUMBER,
                    p_workflow_name IN VARCHAR2,
                    p_workflow_data IN XMLType,
                    p_comment IN VARCHAR2,
                    p_force IN BOOLEAN DEFAULT FALSE) RETURN NUMBER;
```

Table 8–8 lists the parameters that are used in the `WF_IMPORT` function.

**Table 8–8 List of Parameters for WF_IMPORT Function**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_project_id</td>
<td>Specify the ID of the project in to which the workflow will be imported.</td>
</tr>
<tr>
<td>p_workflow_name</td>
<td>Specify the workflow to import.</td>
</tr>
<tr>
<td>p_workflow_data</td>
<td>Specify the workflow meta data. This workflow should be previously exported by the Oracle Data Miner and the workflow version should not be newer than what the Repository supports.</td>
</tr>
<tr>
<td>p_comment</td>
<td>Specify the comment to be applied to the workflow.</td>
</tr>
</tbody>
</table>
| p_force        | Determines whether to force import if the workflow has object name conflicts with existing workflows in the repository. The applicable values are:  
                  - If `p_force` = FALSE, then it raises an exception with a list of conflicting object names.  
                  - If `p_force` = TRUE, then it proceeds with the import of the workflow. |
8.2.9 WF_EXPORT

The WF_EXPORT function exports a specified workflow. If the workflow is either already running or opened by the Oracle Data Miner, then it raises an exception. Alternatively, you can query the ODMR_USER_PROJECT_WORKFLOW for workflows to export.

Function:

FUNCTION WF_EXPORT(p_workflow_id IN NUMBER) RETURN XMLType;

Table 8–9 lists the parameters that are used in the WF_EXPORT function.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_workflow_id</td>
<td>Specify the ID of the workflow to export.</td>
</tr>
</tbody>
</table>

See Also: ODMR_USER_PROJECT_WORKFLOW repository view.

8.3 Repository Views

The Repository Views enable you to:

- Query workflow and project information
- Monitor workflow execution status
- Query generated results

The following repository APIs or views are available:

- ODMR_USER_PROJECT_WORKFLOW
- ODMR_USER_WORKFLOW_ALL
- ODMR_USER_WORKFLOW_LOG
- ODMR_USER_WORKFLOW_NODES
- ODMR_USER_WORKFLOW_MODELS
- ODMR_USER_WF_CLAS_TEST_RESULTS
- ODMR_USER_WF_REGR_TEST_RESULTS
- ODMR_USER_WF_TEST_RESULTS

8.3.1 ODMR_USER_PROJECT_WORKFLOW

You can query all workflows that belong to a specific project or all projects by using the ODMR_USER_PROJECT_WORKFLOW repository view. This view provides information about the workflow, such as status, creation time, update time, and so on. Table 8–10 provides more information about this view.

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROJECT_ID</td>
<td>NUMBER</td>
<td>This is the ID of the project in which the workflow was created in.</td>
</tr>
<tr>
<td>PROJECT_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the project in which the workflow was created in.</td>
</tr>
</tbody>
</table>
8.3.2 ODMR_USER_WORKFLOW_ALL

You can query individual workflow node status after the workflow is complete by using the ODMR_USER_WORKFLOW_ALL repository view. For example, you can query the nodes that failed along with the associated error details, in the last workflow run. Table 8–11 provides more information about this view.

### Table 8–11 ODMR_USER-WORKFLOW_ALL Repository View

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKFLOW_ID</td>
<td>NUMBER</td>
<td>This is the ID of the workflow.</td>
</tr>
<tr>
<td>WF_JOB_NAME</td>
<td>VARCHAR2(261)</td>
<td>This is the Scheduler Job that runs the workflow.</td>
</tr>
<tr>
<td>LOG_DATE</td>
<td>TIMESTAMP(6) WITH TIME ZONE</td>
<td>This is the log entry time stamp.</td>
</tr>
</tbody>
</table>
You can query the logs of workflow run by using the `ODMR_USER_WORKFLOW_LOG` repository view. Oracle Data Miner uses this view to extract and display the workflow event log. Table 8–12 provides more information about this view.

### 8.3.3 ODMR_USER_WORKFLOW_LOG

You can query the logs of workflow run by using the `ODMR_USER_WORKFLOW_LOG` repository view. Oracle Data Miner uses this view to extract and display the workflow event log. Table 8–12 provides more information about this view.

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOG_ID</td>
<td>NUMBER</td>
<td>This is the log entry ID.</td>
</tr>
<tr>
<td>NODE_ID</td>
<td>VARCHAR2(261)</td>
<td>This is the workflow node ID.</td>
</tr>
<tr>
<td>SUBNODE_ID</td>
<td>VARCHAR2(261)</td>
<td>This is the workflow sub node. For example, a Model node in a Build node.</td>
</tr>
<tr>
<td>NODE_STATUS</td>
<td>VARCHAR2(11)</td>
<td>RUNNING: Indicates that the node is running.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SUCCEEDED: Indicates that the node run has completed successfully.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FAILED: Indicates that the node has failed to complete.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOT_STARTED: Indicates that the node is waiting to run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SCHEDULED: Indicates that the node is scheduled to run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PAUSED: Indicates that the node is paused. This is an exception.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STOPPED: Indicates that the node run has stopped.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STALLED: Indicates that the node has stalled. This is an exception.</td>
</tr>
<tr>
<td>SUBNODE_STATUS</td>
<td>VARCHAR2(30)</td>
<td>Same as Node status</td>
</tr>
<tr>
<td>NODE_START_TIME</td>
<td>TIMESTAMP(6) WITH TIME ZONE</td>
<td>This is the workflow node start time stamp.</td>
</tr>
<tr>
<td>NODE_RUN_TIME</td>
<td>INTERVAL DAY(9) TO SECOND(6)</td>
<td>This is the node run time.</td>
</tr>
<tr>
<td>ERROR_CODE</td>
<td>NUMBER</td>
<td>This is the error code returned by the node.</td>
</tr>
<tr>
<td>LOG_MESSAGE</td>
<td>VARCHAR2(4000 CHAR)</td>
<td>This is the log message generated by the node.</td>
</tr>
</tbody>
</table>

### Table 8–12 ODMR_USER_WORKFLOW_LOG Repository Views

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOG_ID</td>
<td>NUMBER</td>
<td>This is the log entry ID.</td>
</tr>
<tr>
<td>JOB_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the Scheduler Job that runs the workflow.</td>
</tr>
<tr>
<td>PROJ_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the project in which the workflow was created in.</td>
</tr>
<tr>
<td>PRO_ID</td>
<td>NUMBER</td>
<td>This is the project ID in which the workflow was created in.</td>
</tr>
<tr>
<td>WF_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the workflow name.</td>
</tr>
<tr>
<td>Column</td>
<td>Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>WF_ID</td>
<td>NUMBER</td>
<td>This is the workflow ID.</td>
</tr>
<tr>
<td>NODE_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the node in the workflow.</td>
</tr>
<tr>
<td>NODE_ID</td>
<td>VARCHAR2(30)</td>
<td>This is the node ID.</td>
</tr>
<tr>
<td>SUBNODE_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the workflow sub node name. For example, Model name in a Build node.</td>
</tr>
<tr>
<td>SUBNODE_ID</td>
<td>VARCHAR2(30)</td>
<td>This is the workflow sub node ID. For example, the Model ID in a Build Node.</td>
</tr>
<tr>
<td>LOG_TIMESTAMP</td>
<td>TIMESTAMP(6) WITH TIME ZONE</td>
<td>This is the log entry time stamp.</td>
</tr>
<tr>
<td>LOG_DURATION</td>
<td>INTERVAL DAY(3) TO SECOND(0)</td>
<td>This is the log entry duration in days and seconds.</td>
</tr>
<tr>
<td>LOG_TYPE</td>
<td>VARCHAR2 (30 CHAR)</td>
<td>WARN: Indicates warning. ERR: Indicates error. INFO: Indicates informational content.</td>
</tr>
<tr>
<td>LOG_SUBTYPE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>START: Indicates start of a task. END: Indicates end of a task.</td>
</tr>
<tr>
<td>LOG_MESSAGE</td>
<td>NVARCHAR2(2000)</td>
<td>This is the log message generated by the node.</td>
</tr>
<tr>
<td>LOG_MESSAGE_DETAILS</td>
<td>VARCHAR2(4000 CHAR)</td>
<td>This is the log message details generated by the node.</td>
</tr>
</tbody>
</table>
8.3.4 ODMR_USER_WORKFLOW_NODES

You can query information about the individual nodes, such as node name, node status, node ID and so on, that are part of a workflow, by using the ODMR_USER_
WORKFLOW_NODES repository view. Table 8–13 provides more information about this view.

Table 8–13  ODMR_USER_WORKFLOW_NODES Repository Views

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROJECT_ID</td>
<td>NUMBER</td>
<td>This is the ID of the project in which the workflow was created in.</td>
</tr>
<tr>
<td>PROJECT_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the project in which the workflow was created in.</td>
</tr>
<tr>
<td>WORKFLOW_ID</td>
<td>NUMBER</td>
<td>This is the ID of the workflow.</td>
</tr>
<tr>
<td>WORKFLOW_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the workflow.</td>
</tr>
<tr>
<td>NODE_TYPE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the node type.</td>
</tr>
<tr>
<td>NODE_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the node.</td>
</tr>
<tr>
<td>NODE_ID</td>
<td>NUMBER</td>
<td>This is the ID of the node.</td>
</tr>
<tr>
<td>NODE_STATUS</td>
<td>VARCHAR2(10 CHAR)</td>
<td>INVALID: Indicates that the node is not valid, and it cannot be run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WARNING: Indicates that the node has run but with warning.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>READY: Indicates that the node is ready to run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FAILURE: Indicates that the node run has failed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COMPLETE: Indicates that the node run has completed successfully.</td>
</tr>
</tbody>
</table>

8.3.5 ODMR_USER_WORKFLOW_MODELS

You can query mining models that belong to a specific Build or Model node of a workflow by using the ODMR_USER_WORKFLOW_MODELS repository view. Table 8–14 provides more information about this view.

Table 8–14  ODMR_USER_WORKFLOW_MODELS Repository Views

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Types</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROJECT_ID</td>
<td>NUMBER</td>
<td>This is the project ID in which the workflow was created in.</td>
</tr>
<tr>
<td>PROJECT_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the project in which the workflow was created in.</td>
</tr>
<tr>
<td>WORKFLOW_ID</td>
<td>NUMBER</td>
<td>This is the ID of the workflow.</td>
</tr>
<tr>
<td>WORKFLOW_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the workflow.</td>
</tr>
</tbody>
</table>
You can query the generated classification results for a specific mining model in the last workflow run by using the `ODMR_USER_WF_CLAS_TEST_RESULTS` repository view. Table 8–15 provides more information about this view.

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NODE_TYPE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the node type.</td>
</tr>
<tr>
<td>NODE_ID</td>
<td>NUMBER</td>
<td>This is the workflow node ID.</td>
</tr>
<tr>
<td>NODE_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the workflow node.</td>
</tr>
<tr>
<td>NODE_STATUS</td>
<td>VARCHAR2(30 CHAR)</td>
<td>INVALID: Indicates that the node is not valid, and cannot be run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WARNING: Indicates that the node has run but with warning.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>READY: Indicates that the node is ready to run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FAILURE: Indicates that the node run has failed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COMPLETE: Indicates that the node run has completed successfully.</td>
</tr>
<tr>
<td>MODEL_TYPE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the model type. For example, Naive Bayes Model.</td>
</tr>
<tr>
<td>MODEL_ID</td>
<td>NUMBER</td>
<td>This is the ID of the model.</td>
</tr>
<tr>
<td>MODEL_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the model.</td>
</tr>
<tr>
<td>MODEL_STATUS</td>
<td>VARCHAR2(30 CHAR)</td>
<td>Same as node status.</td>
</tr>
<tr>
<td>MODEL_CREATIONDATE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the date when the model is created.</td>
</tr>
<tr>
<td>PROJECT_ID</td>
<td>NUMBER</td>
<td>This is the ID of the project in which the workflow was created in.</td>
</tr>
<tr>
<td>PROJECT_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the project in which the workflow was created in.</td>
</tr>
<tr>
<td>WORKFLOW_ID</td>
<td>NUMBER</td>
<td>This is the ID of the workflow.</td>
</tr>
<tr>
<td>WORKFLOW_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the workflow.</td>
</tr>
<tr>
<td>NODE_TYPE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the node type.</td>
</tr>
<tr>
<td>NODE_ID</td>
<td>NUMBER</td>
<td>This is the ID of the node.</td>
</tr>
</tbody>
</table>

8.3.6 ODMR_USER_WF_CLAS_TEST_RESULTS

You can query the generated classification results for a specific mining model in the last workflow run by using the `ODMR_USER_WF_CLAS_TEST_RESULTS` repository view. Table 8–15 provides more information about this view.
<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NODE_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the node.</td>
</tr>
</tbody>
</table>
| NODE_STATUS            | VARCHAR2(10 CHAR) | - INVALID: Indicates that the node is not ready to be run.  
- WARNING: Indicates that the node has run completely, but with warning.  
- READY: Indicates that the node is ready to be run.  
- FAILURE: Indicates that the node run has failed.  
- COMPLETE: Indicates that the node has run successfully. |
| MODEL_ID               | NUMBER          | This is the ID of the model.                                                                                                               |
| MODEL_NAME             | VARCHAR2(30 CHAR) | This is the name of the model.                                                                                                             |
| MODEL_STATUS           | VARCHAR2(10 CHAR) | - WARNING: Indicates that the model has run, but with warning.  
- READY: Indicates that the model is ready to be run.  
- FAILURE: Indicates that the model run has failed.  
- COMPLETE: Indicates that the model run has completed successfully. |
| MODEL_CREATIONDATE     | VARCHAR2(30 CHAR) | This is the creation time of the model.                                                                                   |
| TEST_METRICS           | VARCHAR2(128 CHAR) | The test metric result table that contains Predictive Confidence, accuracy, and so on.                                                 |
| CONFUSION_MATRIX        | VARCHAR2(128 CHAR) | The Confusion Matrix result table.                                                                                                     |
| LIFTS                  | DM_NESTED_CATEGORICALS | The table of DM_NESTED_CATEGORICAL, where:  
  - ATTRIBUTE_NAME contains target class  
  - VALUE contains lift result table |
| ROCS                   | DM_NESTED_CATEGORICALS | The table of DM_NESTED_CATEGORICAL, where:  
  - ATTRIBUTE_NAME contains target class  
  - VALUE contains ROC result table |
8.3.7 ODMR_USER_WF_REGR_TEST_RESULTS

You can query the generated regression results for a specific mining model in the last workflow run by using the `ODMR_USER_WF_REGR_TEST_RESULTS` repository view. Table 8–16 provides more information about this view.

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROJECT_ID</td>
<td>NUMBER</td>
<td>This is the ID of the project under which the workflow is created.</td>
</tr>
<tr>
<td>PROJECT_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the project under which the workflow is created.</td>
</tr>
<tr>
<td>WORKFLOW_ID</td>
<td>NUMBER</td>
<td>This is the workflow ID.</td>
</tr>
<tr>
<td>WORKFLOW_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the workflow.</td>
</tr>
<tr>
<td>NODE_TYPE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the type of the node. For example, Build node, Model node, and so on.</td>
</tr>
<tr>
<td>NODE_ID</td>
<td>NUMBER</td>
<td>This is the ID of the node.</td>
</tr>
<tr>
<td>NODE_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the node.</td>
</tr>
<tr>
<td>NODE_STATUS</td>
<td>VARCHAR2(10 CHAR)</td>
<td>INVALID: Indicates that the node is not ready to be run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WARNING: Indicates that the node has run completely, but with warning.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>READY: Indicates that the node is ready to be run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FAILURE: Indicates that the node run has failed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COMPLETE: Indicates that the node has run successfully.</td>
</tr>
<tr>
<td>MODEL_ID</td>
<td>NUMBER</td>
<td>This is the ID of the model.</td>
</tr>
<tr>
<td>MODEL_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the model.</td>
</tr>
</tbody>
</table>

The table of `DM_NESTED NUMERICALS`, where:

- ATTRIBUTE_NAME contains target class
- VALUE contains ROC area under curve value
Table 8–16  (Cont.) ODMR_USER_WF_REGR_TEST_RESULTS Repository Views

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODEL_STATUS</td>
<td>VARCHAR2(10 CHAR)</td>
<td>WARNING: Indicates that the model has run, but with warning.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>READY: Indicates that the model is ready to be run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FAILURE: Indicates that the model run has failed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COMPLETE: Indicates that the model run has completed successfully.</td>
</tr>
<tr>
<td>MODEL_CREATION_DATE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the creation date of the model.</td>
</tr>
<tr>
<td>TEST_METRICS</td>
<td>VARCHAR2(128 CHAR)</td>
<td>This is the test metrics result table that contains Predictive Confidence, Root Mean Square error, and so on.</td>
</tr>
<tr>
<td>RESIDUAL_PLOT</td>
<td>VARCHAR2(128 CHAR)</td>
<td>This is the test Residual Plot table.</td>
</tr>
</tbody>
</table>

Table 8–16 ODMR_USER_WF_REGR_TEST_RESULTS Repository Views

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROJECT_ID</td>
<td>NUMBER</td>
<td>The project ID of the project under which the workflow is created.</td>
</tr>
<tr>
<td>PROJECT_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>The name of the project under which the workflow is created.</td>
</tr>
<tr>
<td>WORKFLOW_ID</td>
<td>NUMBER</td>
<td>This is the workflow ID.</td>
</tr>
<tr>
<td>WORKFLOW_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the workflow.</td>
</tr>
<tr>
<td>NODE_TYPE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the type of the node. For example, Build node, Model node and so on.</td>
</tr>
<tr>
<td>NODE_ID</td>
<td>NUMBER</td>
<td>This is the ID of the node.</td>
</tr>
<tr>
<td>NODE_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the node.</td>
</tr>
</tbody>
</table>

8.3.8 ODMR_USER_WF_TEST_RESULTS

You can query the combined results of the ODMR_USER_WF_CLAS_TEST_RESULTS and ODMR_USER_WF_REGR_TEST_RESULTS by using the ODMR_USER_WF_TEST_RESULTS repository view. Table 8–17 provides more information about this view.

Table 8–17 ODMR_USER_WF_TEST_RESULTS Repository Views

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROJECT_ID</td>
<td>NUMBER</td>
<td>The project ID of the project under which the workflow is created.</td>
</tr>
<tr>
<td>PROJECT_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>The name of the project under which the workflow is created.</td>
</tr>
<tr>
<td>WORKFLOW_ID</td>
<td>NUMBER</td>
<td>This is the workflow ID.</td>
</tr>
<tr>
<td>WORKFLOW_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the workflow.</td>
</tr>
<tr>
<td>NODE_TYPE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the type of the node. For example, Build node, Model node and so on.</td>
</tr>
<tr>
<td>NODE_ID</td>
<td>NUMBER</td>
<td>This is the ID of the node.</td>
</tr>
<tr>
<td>NODE_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the node.</td>
</tr>
<tr>
<td>Column</td>
<td>Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NODE_STATUS</td>
<td>VARCHAR2(10 CHAR)</td>
<td>INVALID: Indicates that the node is not ready to be run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WARNING: Indicates that the node has run completely, but with warning.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>READY: Indicates that the node is ready to be run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FAILURE: Indicates that the node run has failed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COMPLETE: Indicates that the node has run successfully.</td>
</tr>
<tr>
<td>MODEL_ID</td>
<td>NUMBER</td>
<td>This is the ID of the model.</td>
</tr>
<tr>
<td>MODEL_NAME</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the name of the model.</td>
</tr>
<tr>
<td>MODEL_STATUS</td>
<td>VARCHAR2(10 CHAR)</td>
<td>WARNING: Indicates that the model has run, but with warning.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>READY: Indicates that the model is ready to be run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FAILURE: Indicates that the model run has failed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COMPLETE: Indicates that the model run has completed successfully.</td>
</tr>
<tr>
<td>MODEL_CREATION_DATE</td>
<td>VARCHAR2(30 CHAR)</td>
<td>This is the creation date of the model.</td>
</tr>
<tr>
<td>TEST_METRICS</td>
<td>VARCHAR2(128 CHAR)</td>
<td>This is the test metrics result table that contains Predictive Confidence, Root Mean Square error and so on.</td>
</tr>
<tr>
<td>CONFUSION_MATRIX</td>
<td>VARCHAR2(128 CHAR)</td>
<td>This is the test Confusion Matrix result table.</td>
</tr>
<tr>
<td>LIFTS</td>
<td>DM_NESTED_CATEGORICALS</td>
<td>Table of DM_NESTED_CATEGORICAL, where:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ATTRIBUTE_NAME contains target class</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ VALUE contains lift result table</td>
</tr>
<tr>
<td>ROCS</td>
<td>DM_NESTED_CATEGORICALS</td>
<td>Table of DM_NESTED_CATEGORICAL, where:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ATTRIBUTE_NAME contains target class</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ VALUE contains ROC result table</td>
</tr>
</tbody>
</table>
8.4 PL/SQL APIs Use Cases

This section describes the following use cases:

- **Use Case to Schedule and Run a Build Workflow**: Demonstrates how to schedule the BUILD workflow to run on the last day of each month.

- **Use Case to Schedule and Run an Apply Workflow**: Demonstrates how to schedule the APPLY workflow to run daily.

**See Also:**
- "Premise of the Use Case"
- "Location of Demo Workflow Files"

8.4.1 Premise of the Use Case

The use case is built on two predefined workflows, which are available in the SQL Developer installation location. The two predefined workflow files are:

- `apply_workflow.xml`: Uses the Model node to reference the model built by the `build_workflow`. Then it uses it for scoring.

- `build_workflow.xml`: Builds a Server Vector Machine Classification model, and then stores the model details or coefficients to a table.

**See Also:** "Location of Demo Workflow Files"

8.4.2 Location of Demo Workflow Files

The workflow files `apply_workflow.xml` and `build_workflow.xml` which contain the predefined workflows are available in the SQL Developer installation location at:

<sqldeveloper>\dataminer\demos\workflows.

8.4.3 Use Case to Schedule and Run a Build Workflow

This use case demonstrates how to run a lineage of a workflow. You can run the workflow by using one of the following methods. Both the methods generate the same result, where all the four nodes in the lineage are run. The methods are:

- Select all nodes in the lineage and use the `RERUN_NODE_ONLY` run mode.

- Select the `MODEL_COEFFICIENTS` node and use the `RERUN_NODE_PARENTS` run mode.

The use case, as demonstrated in Example 8–1, does the following:

<table>
<thead>
<tr>
<th>Column</th>
<th>Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROC_AREA</td>
<td>DM_NESTED_NUMERICALS</td>
<td>Table of DM_NESTED_NUMERICALS, where:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ATTRIBUTE_NAME contains target class</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ VALUE contains ROC area under curve value</td>
</tr>
<tr>
<td>RESIDUAL_PLOT</td>
<td>VARCHAR2(128 CHAR)</td>
<td>This is the test Residual Plot table.</td>
</tr>
</tbody>
</table>
Schedules the workflow to run monthly on the last day of the month (BYMONTHDAY=-1) starting at mid night from 12/31/2014 to 12/31/2015 in EST zone.

Executes the WF_RUN API. In this use case, the API WF_RUN with Project Name, Workflow Name Node Name and Time Interval schedules the workflow to run.

Polls the status of the workflow from the ODMR_USER_PROJECT_WORKFLOW view to determine whether the workflow run is complete.

Prints out any node failure from the event log along with error message.

**Example 8–1  Schedule and Run a Workflow, Poll Status and Print Node Failures**

CONNECT DMUSER/DMUSER
SET SERVEROUTPUT ON
DECLARE
  v_jobId VARCHAR2(30) := NULL;
  v_status VARCHAR2(30) := NULL;
  v_projectName VARCHAR2(30) := 'Project';
  v_workflow_name VARCHAR2(30) := 'build_workflow';
  v_node VARCHAR2(30) := 'MODEL_COEFFICIENTS';
  v_run_mode VARCHAR2(30) := ODMRSYS.ODMR_WORKFLOW.RERUN_NODE_PARENTS;
  v_failure NUMBER := 0;
  v_nodes ODMRSYS.ODMR_OBJECT_NAMES := ODMRSYS.ODMR_OBJECT_NAMES();
BEGIN
  v_nodes.extend();
  v_nodes(v_nodes.count) := v_node;
  v_jobId := ODMRSYS.ODMR_WORKFLOW.WF_RUN(p_project_name => v_projectName,
                                             p_workflow_name => v_workflow_name,
                                             p_node_names => v_nodes,
                                             p_run_mode => v_run_mode,
                                             p_start_date => '31-DEC-14 12.00.00 AM AMERICA/NEW_YORK',
                                             p_repeat_interval => 'FREQ=MONTHLY;BYMONTHDAY=-1',
                                             p_end_date => '31-DEC-15 12.00.00 AM AMERICA/NEW_YORK');
  DBMS_OUTPUT.PUT_LINE('Job: '||v_jobId);
  -- wait for workflow to run to completion
  LOOP
    SELECT STATUS INTO v_status FROM ODMR_USER_PROJECT_WORKFLOW
    WHERE WORKFLOW_NAME = v_workflow_name;
    IF (v_status IN ('SCHEDULED', 'ACTIVE')) THEN
      DBMS_LOCK.SLEEP(10); -- wait for 10 secs
    ELSE
      EXIT; -- workflow run completes
    END IF;
  END LOOP;
  -- print all failed nodes from the event log
  FOR wf_log IN (
    SELECT node_id, node_name, subnode_id, subnode_name, log_message, log_message_details
    FROM ODMR_USER_WORKFLOW_LOG
    WHERE job_name=v_jobId and log_type='ERR' and log_message IS NOT NULL)
  LOOP
    DBMS_OUTPUT.PUT_LINE('Node Id: '||wf_log.node_id||', '||'Node Name: '||wf_log.node_name);
    IF (wf_log.subnode_id IS NOT NULL) THEN
      DBMS_OUTPUT.PUT_LINE('Subnode Id: '||wf_log.subnode_id||', '||'Subnode Name: '||wf_log.subnode_name);
    END IF;
  END LOOP;
END;
v_failure := v_failure + 1;
END LOOP;
IF (v_failure = 0) THEN
  DBMS_OUTPUT.PUT_LINE('Workflow Status: SUCCEEDED');
ELSE
  DBMS_OUTPUT.PUT_LINE('Workflow Status: FAILURE');
END IF;
EXCEPTION WHEN OTHERS THEN
  DBMS_OUTPUT.PUT_LINE('Error: '||SUBSTR(DBMS_UTILITY.FORMAT_ERROR_STACK(), 1, 4000));
END;

See Also: ■"WF_RUN"
■ "Querying Named Objects"
■ "Location of Demo Workflow Files"

8.4.3.1 Querying Named Objects
After the workflow run completes successfully, you can query all named objects
generated by the workflow, such as table or view in the Create Table node, models in
the Build nodes, and so on.

The following query returns the model CLAS_SVM_MODEL_2 information.
SELECT * FROM USER_MINING_MODELS WHERE MODEL_NAME = 'CLAS_SVM_MODEL_2
The query result is displayed in Figure 8–1.

Figure 8–1 Query Output for Named Objects

8.4.3.2 Querying the MODEL_COEFFICIENT Table
To query data from the MODEL_COEFFICIENTS table, run the following query:
SELECT * FROM MODEL_COEFFICIENTS
The output of this query is displayed in a table, as shown in Figure 8–2.

Figure 8–2 Query Output for MODEL_COEFFICIENT Table

8.4.3.3 Querying Test Results
You can query the test results from:
■ ODMR_USER_WF_CLAS_TEST_RESULTS Repository View
This section contains examples to query the following:

- Query Test Metrics and Confusion Matrix results: Example 8–2
- Query Test Metrics: Example 8–3
- Query Confusion Matrix: Example 8–4
- Query Lift Table results: Example 8–5

**Example 8–2  Querying Test Metrics and Confusion Matrix Results**

```sql
SELECT TEST_METRICS, CONFUSION_MATRIX FROM ODMR_USER_WF_CLAS_TEST_RESULTS WHERE WORKFLOW_NAME = 'build_workflow' AND NODE_NAME = 'Class Build'
```

The output of this query is shown in the figure below. The query fetches the Test Metrics and Confusion Matrix from the ODMR_USER_WF_CLAS_TEST_RESULTS.

**Example 8–3  Querying TEST_METRICS**

```sql
SELECT * FROM ODMR$18_51_18_106346IFHRNMF
```

The output of this query is shown in the screenshot below. It queries the Test Metrics ODMR$18_51_18_106346IFHRNMF. It fetches the Metric name, the metric VARCHAR value and the metric NUM value.

**Example 8–4  Querying CONFUSION_MATRIX**

```sql
SELECT * FROM ODMR$18_51_17_954530VMUXPWL
```

The output of this query is shown in the screenshot below. It queries the Confusion Matrix ODMR$18_51_17_954530VMUXPWL. It fetches the actual target name, and the predicted target value.

**Example 8–5  Querying Lift Result Table from CLAS_SVM_MODEL_2**

```sql
SELECT MODEL_NAME, a.ATTRIBUTE_NAME "target value", a.VALUE "lift result table"
```
PL/SQL APIs Use Cases

FROM
ODMR_USER_WF_CLAS_TEST_RESULTS, TABLE(LIFTS) a
WHERE
WORKFLOW_NAME = 'build_workflow' AND NODE_NAME = 'Class Build' AND ATTRIBUTE_NAME='Yes'

The output of this query is shown in the screenshot below. It queries the Lift Result table from the CLAS_SVM_MODEL_2.

8.4.4 Use case to Schedule and Run an Apply Workflow

This use case demonstrates how to run a lineage of the APPLY workflow. To run the lineage, specify the INSUR_CUST_LTV_SAMPLE APPLY node and use the RERUN_NODE_CHILDREN run mode.

The use case, as demonstrated in Example 8–6, does the following:

- Schedules the APPLY workflow to run daily from mid night of 12/31/2014 to 12/31/2015 in EST zone
- Executes the WF_RUN API
- Polls the status of the workflow from the ODMR_USER_PROJECT_WORKFLOW view
- Prints the failed nodes

Example 8–6  Schedule and Run the Apply Workflow, Poll Status and Print Node Failures

CONNECT DMUSER/DMUSER
SET SERVEROUTPUT ON
DECLARE
    v_jobId VARCHAR2(30) := NULL;
    v_status VARCHAR2(30) := NULL;
    v_projectName VARCHAR2(30) := 'Project';
    v_workflow_name VARCHAR2(30) := 'apply_workflow';
    v_node VARCHAR2(30) := 'INSUR_CUST_LTV_SAMPLE APPLY';
    v_run_mode VARCHAR2(30) := ODMRSYS.ODMR_WORKFLOW.RERUN_NODE_CHILDREN;
    v_failure NUMBER := 0;
    v_nodes ODMRSYS.ODMR_OBJECT_NAMES := ODMRSYS.ODMR_OBJECT_NAMES();
BEGIN
    v_nodes.extend();
    v_nodes(v_nodes.count) := v_node;
    v_jobId := ODMRSYS.ODMR_WORKFLOW.WF_RUN(p_project_name => v_projectName,
        p_workflow_name => v_workflow_name,
        p_node_names => v_nodes,
        p_run_mode => v_run_mode,
        p_start_date => '31-DEC-14 12.00.00 AM AMERICA/NEW_YORK',
        p_repeat_interval => 'FREQ= DAILY',
        p_end_date => '31-DEC-15 12.00.00 AM AMERICA/NEW_YORK');
    DBMS_OUTPUT.PUT_LINE('Job: '||v_jobId);
    -- wait for workflow to run to completion
    LOOP
        SELECT STATUS INTO v_status FROM ODMR_USER_PROJECT_WORKFLOW
            WHERE WORKFLOW_NAME = v_workflow_name;
        IF (v_status IN ('SCHEDULED', 'ACTIVE')) THEN
            DBMS_LOCK.SLEEP(10); -- wait for 10 secs
        ELSE
            EXIT; -- workflow run completes
        END IF;
    END LOOP;
END;
/
END IF;
END LOOP;
-- print all failed nodes (see example above)
EXCEPTION WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error: '||SUBSTR(DBMS_UTILITY.FORMAT_ERROR_STACK(), 1, 4000));
END;

See Also: ■ WF_RUN"
 ■ "Premise of the Use Case"
 ■ "Location of Demo Workflow Files"
 ■ "Querying Scoring Result"

8.4.4.1 Querying Scoring Result
After the workflow run completes successfully, you can query the scoring result directly.

To query the scoring results:

SELECT * FROM SCORED_CUSTOMERS_TBL

The output of the query is displayed in a table, as shown in Figure 8–3.

Figure 8–3 Query Output for Scoring Results
The Oracle Data Miner client must be compatible with the Oracle Data Miner repository in Oracle Database. When the versions are incompatible, Oracle Data Miner displays an error message that provides information to assist you in upgrading either the client or the repository.

For example, if a newer client attempts to connect to an older repository, then the error shown in Figure A–1 is displayed.

**Figure A–1  Older Oracle Data Miner Repository Error**

![Older Oracle Data Miner Repository Error]

If an older client attempts to connect to a new repository, then the error shown in Figure A–2 is displayed.

**Figure A–2  Newer Oracle Data Miner Repository Error**

![Newer Oracle Data Miner Repository Error]

When you evaluate how to address the errors shown in Figure A–1 and Figure A–2, you must take several version numbers into account. The version number combinations that are supported in each release are listed in Table A–1. The version numbers shown in the columns of the table are described as follows:

- **SQL Developer** — This is the information that is displayed when you select the SQL Developer Help menu, then the **About** tab.
■ **Oracle Data Miner Extension** — This is the Oracle Data Miner version that is displayed when you select the SQL Developer Help menu, then the **Extensions** tab.

■ **Oracle Data Miner Repository Version** — You can find the repository version by running the query shown in Section 5.2, "Determining the Status and Version of the Repository".

■ **Oracle Data Miner XML Schema Version** — This is the version of the XML schema used by the Oracle Data Miner workflows in the repository. This number is shown when you export or import workflows.

**Note:** The SQL Developer version and the Oracle Data Miner Extension version uniquely identify the client. The Oracle Data Miner Repository Version uniquely identifies the repository.
<table>
<thead>
<tr>
<th>SQL Developer</th>
<th>Data Miner Extension</th>
<th>Data Miner Repository Version</th>
<th>Data Miner XML Schema Version</th>
<th>Release Type (Full or Path)</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL Developer 3.0 EA 4 3.0.03 Build MAIN-03.97</td>
<td>11.2.0.1.9.96</td>
<td>11.2.0.1.9</td>
<td>11.2.0.1.9</td>
<td>EA4 (only EA release to include Data Miner)</td>
<td>Does not work on 11.2.0.3 and above</td>
</tr>
<tr>
<td>SQL Developer 3.0 RTM 3.0.04 Build MAIN-04.34</td>
<td>11.2.0.1.10.109</td>
<td>11.2.0.1.10</td>
<td>11.2.0.1.9</td>
<td>RTM (released 03/30/11)</td>
<td>Does not work on 11.2.0.3 or higher</td>
</tr>
<tr>
<td>SQL Developer 3.0</td>
<td>11.2.0.2.04.38</td>
<td>11.2.0.1.12</td>
<td>112.0.1.9</td>
<td>Patch (Dev internal only)</td>
<td>Does not work on 11.2.0.3 or higher</td>
</tr>
<tr>
<td>SQL Developer 3.0</td>
<td>11.2.0.2.04.39</td>
<td>11.2.0.1.13</td>
<td>11.2.0.1.9</td>
<td>Patch (internal)</td>
<td>Does not work on 11.2.0.3 or higher</td>
</tr>
<tr>
<td>SQL Developer 3.0</td>
<td>11.2.0.2.04.40</td>
<td>11.2.0.1.13</td>
<td>11.2.0.1.9</td>
<td>Patch (external) link to patch</td>
<td>Does not work on 11.2.0.3 or higher</td>
</tr>
<tr>
<td>SQL Developer 3.1 EA 1</td>
<td>11.2.1.1.xx.xx</td>
<td>11.2.1.1.1</td>
<td>11.2.1.1.1</td>
<td>EA1 candidate (not released)</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 3.1 EA 1 3.1.05 Build MAIN-05.97</td>
<td>11.2.1.1.05.97</td>
<td>11.2.1.1.2</td>
<td>11.2.1.1.1</td>
<td>EA1 (released 10/10/11)</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 3.1</td>
<td>11.2.1.1.06.44</td>
<td>11.2.1.1.3</td>
<td>11.2.1.1.1</td>
<td>EA2 (released 11/15/11)</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 3.1</td>
<td>11.2.1.1.06.82</td>
<td>11.2.1.1.4</td>
<td>11.2.1.1.1</td>
<td>EA3 (released 12/21/11)</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 3.1 RTM 3.1.07 Build MAIN-07.42</td>
<td>11.2.1.1.07.42</td>
<td>11.2.1.1.5</td>
<td>11.2.1.1.1</td>
<td>RTM (released 02/07/12)</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 3.2 RTM Version 3.2.09 Build MAIN-09.18</td>
<td>11.2.1.1.09.18</td>
<td>11.2.1.1.6</td>
<td>11.2.1.1.1</td>
<td>RTM (released 08/20/12)</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 3.2 RTM Version 3.2.10.09 Build MAIN-09.57</td>
<td>11.2.1.1.09.57</td>
<td>11.2.1.1.6</td>
<td>11.2.1.1.1</td>
<td>RTM (released 09/24/12). For Data Miner, there are no changes made between 3.2 and 3.2.1.</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 3.2.2 RTM Version 3.2.20.09 Build MAIN-09.87</td>
<td>11.2.1.1.09.87</td>
<td>11.2.1.6</td>
<td>11.2.1.1.1</td>
<td>RTM (released 11/01/12). For Data Miner, there are no changes made between 3.2 and 3.2.2.</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 4.0 EA 1 (was 3.3) Version 4.0.0.12 Build MAIN-12.27</td>
<td>11.2.2.1.12.27</td>
<td>11.2.2.1.1</td>
<td>11.2.2.1.1</td>
<td>EA1 (released 07/11/13)</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 4.0 EA 2 Version 4.0.0.12 Build MAIN-12.84</td>
<td>12.2.0.12.84</td>
<td>12.1.0.1.1</td>
<td>12.1.0.1.1</td>
<td>EA2 (released 09/13/13)</td>
<td></td>
</tr>
<tr>
<td>SQL Developer 4.0 EA 3 Version 4.0.0.13 Build MAIN-13.30</td>
<td>12.2.0.13.30</td>
<td>12.1.0.1.2</td>
<td>12.1.0.1.2</td>
<td>EA3 (released 11/05/13)</td>
<td>Migration from prior versions is broken. Workaround documented in OTN posting.</td>
</tr>
</tbody>
</table>

### Table A–1 SQL Developer Oracle Data Miner Extension and Oracle Data Miner Repository Compatibility
<table>
<thead>
<tr>
<th>SQL Developer</th>
<th>Data Miner Extension</th>
<th>Data Miner Repository Version</th>
<th>Data Miner XML Schema Version</th>
<th>Release Type (Full or Path)</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL Developer 4.0 RTM Version</td>
<td>12.2.0.13.80</td>
<td>12.1.0.1.3</td>
<td>12.1.0.1.3</td>
<td>RTM</td>
<td>(12/12/13)</td>
</tr>
<tr>
<td>4.0.0.13 Build MAIN-13.80</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQL Developer 4.0.1 RTM Version</td>
<td>12.2.0.14.48</td>
<td>12.1.0.1.4</td>
<td>12.1.0.1.4</td>
<td>RTM</td>
<td>Data Miner fixes and some minor enhancements</td>
</tr>
<tr>
<td>4.0.1.14 Build MAIN14.48</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQL Developer 4.0.2 Version</td>
<td>12.2.0.15.21</td>
<td>12.1.0.1.5</td>
<td>12.1.0.1.4</td>
<td>RTM</td>
<td>Only critical Data Miner fixes were back ported. There was a change to PL/SQL code, so the repository version is changed. The XML schema was not changed, so the XML workflow version is unchanged. This will allow users to import workflows between 4.0.1 and 4.0.2 with no issues.</td>
</tr>
<tr>
<td>4.0.2.15.21 Build 15.21</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQL Developer 4.0.3 Version</td>
<td>12.2.0.16.84</td>
<td>12.1.0.1.5</td>
<td>12.1.0.1.4</td>
<td>RTM</td>
<td>Changes have been made to Data Miner in the 4.0.3 release, but they do not require a server-side change, so the repository and workflow version numbers remain the same.</td>
</tr>
<tr>
<td>4.0.3.16 Build MAIN-16.84</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQL Developer 4.1 EA1 Version</td>
<td>12.2.0.17.29</td>
<td>12.1.0.2.1</td>
<td>12.1.0.2.1</td>
<td>EA1</td>
<td>(12/09/14)</td>
</tr>
<tr>
<td>4.1.0.17 Build MAIN-17.29</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQL Developer 4.1 EA2 Version</td>
<td>12.2.0.18.37</td>
<td>12.1.0.2.2</td>
<td>12.1.0.2.2</td>
<td>EA2</td>
<td>(03/09/15)</td>
</tr>
<tr>
<td>4.1.0.18 Build MAIN-18.37</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQL Developer 4.1 RTM Version</td>
<td>12.2.0.19.07</td>
<td>12.1.0.2.3</td>
<td>12.1.0.2.3</td>
<td>RTM</td>
<td>Not allowed to be installed on Oracle Database 12.2.</td>
</tr>
<tr>
<td>4.1.0.19 Build MAIN-19.07</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Index</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| **A** | AL32UTF8 character set, 2-2  
ASM, 5-7  
ASSM, 2-3, 5-6, 5-7 |
| **C** | creatxmlworkflowsbackup2.sql, 5-3 |
| **D** | data  
access to, 4-6  
multi-byte, 2-2  
sample, 3-3  
Data Miner  
administration scripts, 5-1  
arquitectura, 1-1  
big data, 1-2  
installation, 3-1  
resource plan for, 6-2 |
| **I** | installation, 1-2  
Data Miner, 3-1  
JSON parser and Data Guide, 3-6  
Oracle R Enterprise, 1-3  
requirements, 2-1  
SQL Developer, 3-2  
internal tables, 4-1 |
| **L** | Linux, 2-3  
locks  
session, 5-6, 5-7, 5-8  
workflows, 4-4 |
| **M** | managing system resources  
for model builds, 6-3  
for parallel execution, 6-5  
for workflow execution, 6-4  
for workloads, 6-1  
MAX_NUM_THREADS, 6-3, 6-6  
MAX_THREAD_WAIT, 6-6  
Microsoft Windows, 2-3  
migratebinaryodmr.sql, 5-5, 5-7  
migrateodmr.sql, 5-5, 5-6 |
| **O** | ODMR_USER_PROJECT_WORKFLOW repository view, 8-7  
ODMR_USER WF_CLAS_TEST_RESULTS repository view, 8-13  
ODMR_USER WF_REGR_TEST_RESULTS repository view, 8-15  
ODMR_USER WF_TEST_RESULTS repository view, 8-16  
ODMR_USER_WORKFLOW_ALL repository view, 8-8  
ODMR_USER_WORKFLOW_LOG repository view, 8-9  
ODMR_USER_WORKFLOW_MODELS repository view, 8-12  
ODMR_USER_WORKFLOW_NODES repository view, 8-11  
ODMRS tables, 4-2  
ODMRSWORKFLOWS_BACKUP able, 5-3  
ODMRSYS, 5-2, 5-5  
See repository  
ODMRUSER role, 5-6, 5-7, 5-8  
Oracle Advanced Analytics, 1-3, 2-1  
Oracle Automatic Segment Space Management See ASM |
Oracle Data Mining, 1-2
  APIs, 1-3
data dictionary views, 1-5
PL/SQL packages, 1-4
SQL scoring functions, 1-4
Oracle Database Examples, 2-2
Oracle Database Recovery Manager
  See RMAN
Oracle Database Resource Manager, 6-1
Oracle Database Scheduler
  See Scheduler
Oracle R Enterprise, 1-2, 1-3, 2-2
Oracle Recovery Manager, 5-2
Oracle Scheduler, 1-2
Oracle Solaris, 2-3
Oracle Text, 1-2, 2-1, 2-2
Oracle Text Knowledge Base, 2-2

P
parallel execution, 6-5
parallel query, 6-5
PARALLEL_DEGREE_POLICY=AUTO, 6-5
PARALLEL_DEGREE_POLICY=INVALID, 6-5
PARALLEL_QUERY_ON_ALLOWED, 6-5, 6-6
PL/SQL APIs
  PROJECT_CREATE, 8-1
  PROJECT_DELETE, 8-1
  PROJECT_RENAME, 8-1
  WF_DELETE, 8-1
  WF_EXPORT, 8-1
  WF_IMPORT, 8-1
  WF_RENAME, 8-1
  WF_RUN, 8-1
  WF_STOP, 8-1
predictive query, 1-4
PROJECT_CREATE
  PL/SQL API, 8-2
PROJECT_DELETE
  PL/SQL API, 8-2
PROJECT_RENAME
  PL/SQL API, 8-2
proxy users, 4-2

R
repository
  about, 1-1, 1-2
  access rights to, 4-5
api, 8-1
backing up and restoring, 5-2
binary storage, 5-6, 5-7
determining the status and version of, 5-1
dropping, 5-8
installation of, 3-1
installing using the GUI, 3-4
installing with a script, 3-2
migrating, 5-5
object-relational storage, 5-6, 5-7
objects in, 4-1
upgrading, 4-1
views, 5-5
repository view
  ODMR_USER_PROJECT_WORKFLOW, 8-7
  ODMR_USER_WF_CLAS_TEST_RESULTS, 8-13
  ODMR_USER_WF_REGR_TEST_RESULTS, 8-15
  ODMR_USER_WF_TEST_RESULTS, 8-16
  ODMR_USER_WORKFLOW_ALL, 8-8
  ODMR_USER_WORKFLOW_LOG, 8-9
  ODMR_USER_WORKFLOW_MODELS, 8-12
  ODMR_USER_WORKFLOW_NODES, 8-12
resource plan, 6-2
restorexmlworkflowfrombackup2.sql, 5-3
RMAN, 5-2, 6-1

S
sample data, 3-3
sample schemas, 2-2
Scheduler, 6-4, 6-6
scoring, 1-4
snippets, 1-4, 1-5
SQL Data Mining functions, 1-4
SQL Developer
  about, 1-1
  installing, 3-2
SQL script generation, 7-1
  use case, 7-1

T
THREAD_WAIT_TIME, 6-6

U
upgradeRepoFromORtoBinary.sql, 5-5, 5-7
upgradeRepoWithNewTableSpace.sql, 5-5, 5-6
users
  Data Miner, 4-1
  multiple user access to repository, 4-4
  objects in user schemas, 4-1
  proxy, 4-2
  shared access to repository, 4-4
  single user access to repository, 4-4

W
WF_DELETE
  PL/SQL API, 8-6
WF_EXPORT
  PL/SQL API, 8-7
WF_IMPORT
  PL/SQL API, 8-6
WF_RENAME
  PL/SQL API, 8-5
WF_RUN
  PL/SQL API, 8-3, 8-4
WF_STOP
  PL/SQL API, 8-5
WORKFLOW_JOB_CLASS, 6-4, 6-6
workflows, 1-1, 1-2, 4-1
backing up, 5-3
restoring from backup, 5-3, 5-4
scheduling, 6-4

X

XML
  binary storage, 2-1, 2-3, 5-6, 5-7
  object-relational storage, 2-1, 2-3, 5-6, 5-7
XML DB, 1-2, 2-1, 2-2
XMLType, 2-3