

Oracle® Data Miner

Installation and Administration Guide



Release 18.4

F12975-01

January 2019

The Oracle logo, consisting of a solid red square with the word "ORACLE" in white, uppercase, sans-serif font centered within it.

ORACLE®

Oracle Data Miner Installation and Administration Guide, Release 18.4

F12975-01

Copyright © 2015, 2019, Oracle and/or its affiliates. All rights reserved.

Primary Author: Moitreyee Hazarika

Contributors: Alex Sakharov, Ari Mozes, Charlie Berger, Denny Wong, Dongfang Bai, Eros Espinola, Jim Dadashev, Gloria Castillo, Mark McCracken, Mark Kelly, Marat Spivak, Pedro Reta, Zabdiel Jaramillo Roman

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

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

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

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

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

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

| | |
|-----------------------------|------|
| Audience | vii |
| Documentation Accessibility | vii |
| Related Documents | vii |
| Conventions | viii |

1 Oracle Data Miner System Overview

| | |
|---|-----|
| Oracle Data Miner Architecture | 1-1 |
| About the Oracle Data Miner Repository | 1-2 |
| Database Features Used by Oracle Data Miner | 1-2 |
| Oracle Data Miner and Oracle Advanced Analytics | 1-3 |
| About Data Mining APIs | 1-4 |
| Data Mining PL/SQL Packages | 1-4 |
| Data Mining SQL Scoring Functions | 1-5 |
| Data Mining Data Dictionary Views | 1-6 |
| Resources For Learning About Oracle Data Miner | 1-6 |

2 Installing the Database to Support Oracle Data Miner

| | |
|---|-----|
| Database Requirements for Oracle Data Miner | 2-1 |
| Oracle Text and Oracle Data Miner | 2-2 |
| XML DB and Oracle Data Miner | 2-3 |
| Storage Configuration for Oracle Data Miner | 2-3 |
| Installing Oracle Database 11.2.0.4 or Later | 2-3 |
| Connecting to an Oracle Cloud Database Schema | 2-4 |

3 Installing Oracle Data Miner

| | |
|--|-----|
| Oracle Data Miner Installation Overview | 3-1 |
| About Oracle Data Miner Sample Data | 3-2 |
| Downloading SQL Developer | 3-3 |
| Installing the Repository Using a Script | 3-3 |

| | |
|---|------|
| Loading the Sample Data Using a Script | 3-5 |
| Installing the Repository Using SQL Developer GUI | 3-6 |
| Installing JSON Parser and Data Guide | 3-8 |
| About the Scripts for Installing JSON Support | 3-9 |
| Requirements for Running the Scripts to Install JSON Parser | 3-9 |
| Installing JSON Parser in a Remote Database | 3-10 |
| Installing JSON Parser Locally on the Database Host | 3-10 |

4 Managing Oracle Data Miner Users

| | |
|--|-----|
| About User Objects and Repository Objects | 4-1 |
| Controlling the Size of Users' Schemas | 4-2 |
| Objects in Oracle Data Miner Users' Schema | 4-2 |
| About Oracle Data Miner Internal Tables | 4-2 |
| About Proxy Users for Oracle Data Miner | 4-3 |
| Setting the JVM Property | 4-5 |
| Choosing an Access Model for Oracle Data Miner | 4-5 |
| Single User Access | 4-6 |
| Multiple User Access | 4-6 |
| Shared User Environment | 4-6 |
| About the Document in Use Condition | 4-7 |
| Granting or Dropping Access Rights to Oracle Data Miner Repository | 4-7 |
| Granting Access Rights Using the GUI | 4-7 |
| Granting Access Rights Using a Script | 4-8 |
| Dropping Access Rights Using a Script | 4-8 |
| Granting Access to Data | 4-9 |

5 Managing the Oracle Data Miner Repository

| | |
|--|-----|
| About Oracle Data Miner Administration Scripts | 5-1 |
| Setting the Path to Oracle Data Miner Scripts | 5-2 |
| Determining the Status and Version of the Repository | 5-2 |
| Backing Up and Restoring the Repository | 5-2 |
| Full Backup and Restore | 5-3 |
| Full Backup and Restore in Database 11.2.0.4 and Later | 5-3 |
| Workflow Only Backup | 5-3 |
| Workflow Only Restore | 5-4 |
| Workflow Only Restore Examples | 5-5 |
| Migrating the Repository | 5-6 |
| Upgrading ODMRSYS | 5-7 |
| Scripts to Migrate the Repository | 5-7 |

| | |
|---|-----|
| Upgrading New ODMRSYS Tablespace From Object-Relational to Binary | 5-8 |
| Upgrading ODMRSYS From Object-Relational to Binary | 5-9 |
| Dropping the Repository | 5-9 |

6 Managing System Resources for Oracle Data Miner

| | |
|--|-----|
| Oracle Data Miner Resource Management Overview | 6-1 |
| Allocating Resources to Oracle Data Miner User Sessions | 6-2 |
| Managing Model Builds | 6-3 |
| Managing Workflow Execution | 6-4 |
| Managing Parallel Processing | 6-5 |
| Summary of Oracle Data Miner Repository Properties for System Management | 6-6 |

7 Generating and Deploying SQL Scripts

| | |
|--|------|
| Overview of the SQL Script Generation Feature | 7-1 |
| Overview of the SQL Script Generation Use Case | 7-1 |
| Premise of the SQL Script Generation Use Case | 7-2 |
| Location of Demo Workflow Files | 7-3 |
| About the Sample Workflow | 7-3 |
| Performing Prerequisite Tasks | 7-4 |
| Importing and Running a Workflow | 7-4 |
| Generating SQL Script Files from the Workflow | 7-6 |
| List of Generated SQL Script Files | 7-9 |
| Variable Definitions in Script Files | 7-10 |
| Control Tables | 7-11 |
| Structure of the Control Table | 7-11 |
| Columns in the Control Table | 7-12 |
| Scheduling Workflow Script Files | 7-13 |
| Prerequisites for Scheduling Workflow Script Files | 7-14 |
| Adding Complete Directory Path in the Master Script | 7-14 |
| Creating Credentials for Database Host and Database | 7-15 |
| Schedule SQL Scripts Using SQL Developer | 7-16 |
| Creating Credentials for Database Host and Database | 7-16 |
| Defining Scheduler Job using Job Wizard | 7-17 |
| Scheduling SQL Scripts using Oracle Enterprise Manager | 7-18 |
| Deploying SQL Scripts on the Target Database | 7-18 |
| Querying the Control Table | 7-19 |

8 Using PL/SQL API to Manage Workflows

| | |
|-------------------|-----|
| About PL/SQL APIs | 8-1 |
|-------------------|-----|

| | |
|---|------|
| PL/SQL APIs | 8-1 |
| PROJECT_CREATE | 8-2 |
| PROJECT_RENAME | 8-3 |
| PROJECT_DELETE | 8-3 |
| WF_RUN | 8-3 |
| WF_RUN Parameters | 8-4 |
| WF_RUN with Project Name, Workflow Name, and Node Name | 8-5 |
| WF_RUN with Project Name, Workflow Name Node Name and Time Interval | 8-6 |
| WF_RUN with Project ID, Workflow ID, Node ID and Time Interval | 8-6 |
| WF_RUN with Project ID, Workflow ID and Node IDs | 8-6 |
| WF_STOP | 8-7 |
| WF_RENAME | 8-7 |
| WF_DELETE | 8-8 |
| WF_IMPORT | 8-8 |
| WF_EXPORT | 8-9 |
| Repository Views | 8-9 |
| ODMR_USER_PROJECT_WORKFLOW | 8-10 |
| ODMR_USER_WORKFLOW_ALL | 8-11 |
| ODMR_USER_WORKFLOW_LOG | 8-12 |
| ODMR_USER_WORKFLOW_NODES | 8-14 |
| ODMR_USER_WORKFLOW_MODELS | 8-15 |
| ODMR_USER_WF_CLAS_TEST_RESULTS | 8-16 |
| ODMR_USER_WF_REGR_TEST_RESULTS | 8-18 |
| ODMR_USER_WF_TEST_RESULTS | 8-19 |
| ODMR_USER_WORKFLOW_ALL_POLL | 8-21 |
| PL/SQL APIs Use Cases | 8-22 |
| Premise of the PL/SQL Use Cases | 8-22 |
| Use case to Schedule and Run Apply Workflows | 8-23 |
| Querying Scoring Result | 8-24 |
| Use Case to Schedule and Run a Build Workflow | 8-24 |
| Querying the MODEL_COEFFICIENT Table | 8-27 |
| Querying Named Objects | 8-27 |
| Querying Test Results | 8-28 |

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](#)
- [Documentation Accessibility](#)
- [Related Documents](#)
- [Conventions](#)

Audience

This document is intended for database administrators and database developers.

Documentation Accessibility

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

Access to Oracle Support

Oracle customers 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> or visit <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 Release 2 (12.2)

- *Oracle Data Miner Release Notes*
- *Oracle Data Miner User's Guide*
- *Oracle SQL Developer User's Guide*
- *Oracle Data Mining Concepts*
- *Oracle Data Mining User's Guide*
- *Oracle R Enterprise User's Guide*

Oracle Database 12c Release 1 (12.1)

- [Oracle Data Miner Release Notes](#)
- [Oracle Data Miner User's Guide](#)
- [Oracle SQL Developer User's Guide](#)
- [Oracle Data Mining Concepts](#)
- [Oracle Data Mining User's Guide](#)
- [Oracle R Enterprise User's Guide](#)

Oracle Database 11g Release 2 (11.2)

- [Oracle SQL Developer User's Guide](#)
- [Oracle Data Mining Concepts](#)
- [Oracle Data Mining User's Guide](#)
- [Oracle Data Mining API Guide \(Virtual Book\)](#)
- [Oracle R Enterprise Installation and Administration Guide](#)
- [Oracle R Enterprise User's Guide](#)

Conventions

The following text conventions are used in this document:

| Convention | Meaning |
|-----------------|--|
| boldface | Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary. |
| <i>italic</i> | Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values. |
| monospace | Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter. |

1

Oracle Data Miner System Overview

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.

- [Oracle Data Miner Architecture](#)
Oracle Data Miner is an extension of Oracle SQL Developer, a graphical development environment for Oracle SQL.
- [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`.
- [Database Features Used by Oracle Data Miner](#)
Oracle Data Miner uses a number of Oracle Database features such as Oracle Data Mining, Oracle XML DB, Oracle R Enterprise and so on.
- [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.
- [About Data Mining APIs](#)
Oracle Data Miner is an application based on the Data Mining APIs in Oracle Database.
- [Resources For Learning About Oracle Data Miner](#)
This section lists the resources such as documentation, forums, blogs, trainings, and tutorials for Oracle Data Miner.

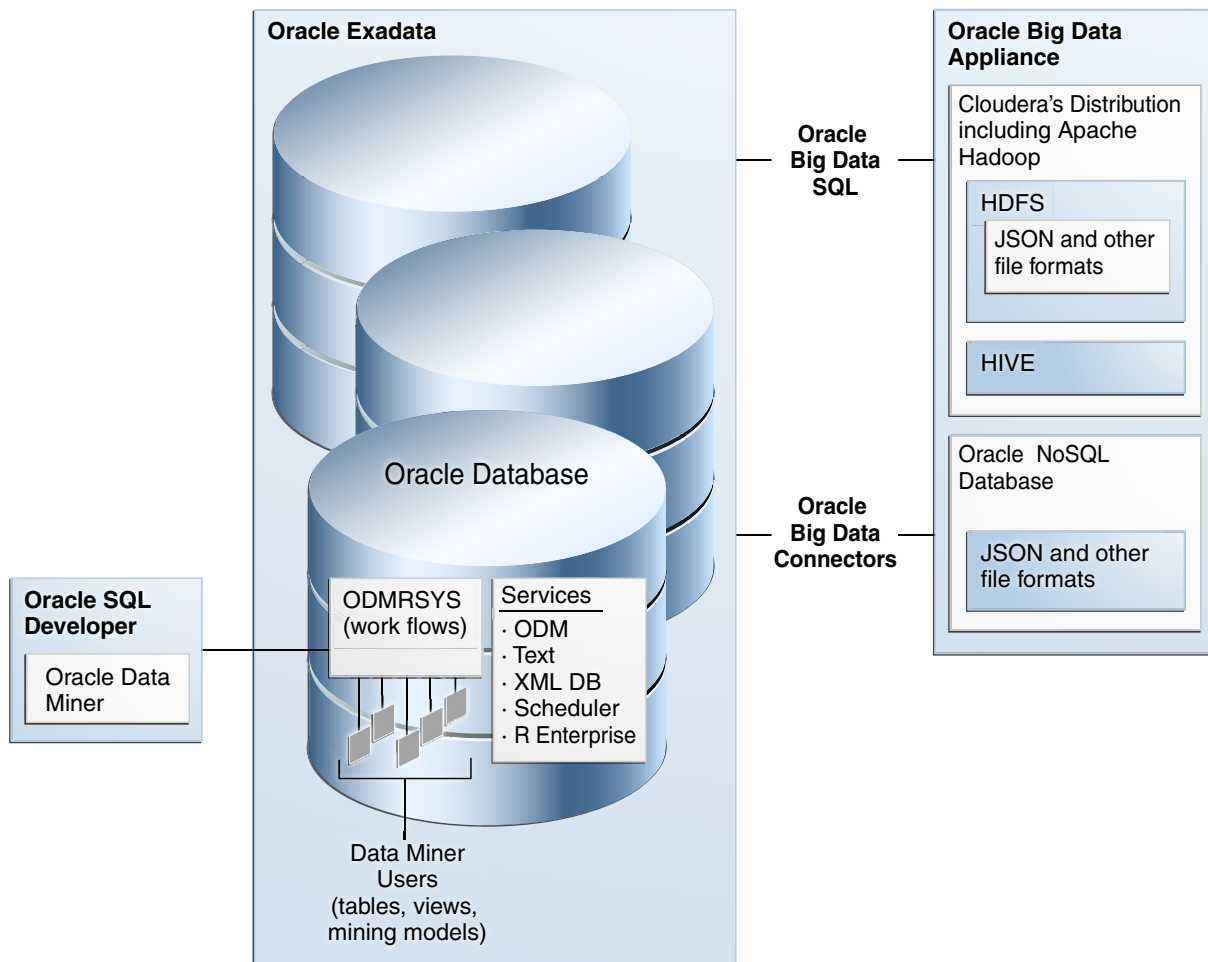
Oracle Data Miner Architecture

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. It uses the `ODMRSYS` schema as a dedicated system repository.

The architecture of Oracle Data Miner is illustrated in [Figure 1-1](#).

Figure 1-1 Oracle Data Miner Architecture for Big Data



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.

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.

Database Features Used by Oracle Data Miner

Oracle Data Miner uses a number of Oracle Database features such as Oracle Data Mining, Oracle XML DB, Oracle R Enterprise and so on.

Oracle Data Miner uses the following Oracle Database features:

- 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.

 **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.

Related Topics

- [Oracle XML DB Developer's Guide](#)
- [Oracle Text Application Developer's Guide](#)
- [Oracle Database Administrator's Guide](#)
- [Oracle R Enterprise Installation and Administration Guide](#)

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.

Related Topics

- [Oracle Data Mining Concepts](#)
- [Oracle R Enterprise User's Guide](#)

About Data Mining APIs

Oracle Data Miner is an application based on the Data Mining APIs in Oracle Database.

The APIs are public and can be used directly for application development. The APIs are summarized in the following topics:

- [Data Mining PL/SQL Packages](#)
PL/SQL APIs manipulate mining models, which are database schema objects.
- [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.
- [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.

Data Mining PL/SQL Packages

PL/SQL APIs manipulate mining models, which are database schema objects.

[Table 1-1](#) lists the PL/SQL packages and their descriptions.

Table 1-1 Oracle Data Mining PL/SQL Packages

| Package | Description |
|----------------------------|---|
| DBMS_DATA_MINING | 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. |
| DBMS_DATA_MINING_TRANSFORM | 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. |
| DBMS_PREDICTIVE_ANALYTICS | Procedures that perform predict, explain, and profile operations without a user-created mining model. |

 **Note:**

The mining operations in the `DBMS_PREDICTIVE_ANALYTICS` package are available in code snippets in Oracle Data Miner.

Related Topics

- *Oracle Data Miner User's Guide*
- *Oracle Database PL/SQL Packages and Types Reference*

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, where the algorithmic processing is performed dynamically without a user-supplied mining model. The term **Predictive Query** refers to this mode of scoring.

Table 1-2 Data Mining SQL Scoring Functions

| Function Name | Function Description |
|------------------------|---|
| CLUSTER_DETAILS | Returns cluster details for each row in the input data. |
| CLUSTER_DISTANCE | Returns the distance between each row and the centroid. |
| CLUSTER_ID | Returns the ID of the highest probability cluster for each row. |
| CLUSTER_PROBABILITY | Returns the highest probability cluster for each row. |
| CLUSTER_SET | Returns a set of cluster ID and probability pairs for each row. |
| FEATURE_COMPARE | Compares two different documents including short ones such as keyword phrases or two attribute lists for similarity or dissimilarity. |
| FEATURE_DETAILS | Returns a set of feature and value pairs for each row. |
| FEATURE_ID | Returns feature details for each row in the input data. |
| FEATURE_SET | Returns a set of feature ID and feature value pairs for each row. |
| FEATURE_VALUE | Returns the value of the highest value feature for each row |
| PREDICTION | Returns the prediction for each row in the input. |
| PREDICTION_BOUNDS | Returns the upper and lower bounds of prediction for each row (GLM only). |
| PREDICTION_COST | Returns a cost for each row. |
| PREDICTION_DETAILS | Returns prediction details for each row. |
| PREDICTION_PROBABILITY | Returns the probability of each prediction. |
| PREDICTION_SET | Returns the prediction or cost with probability for each row. |

**Note:**

The SQL scoring functions are available in code snippets in Oracle Data Miner.

Related Topics

- *Oracle Data Miner User's Guide*
- *Oracle Database SQL Language Reference*
- *Oracle Data Mining User's Guide*

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 1-3 Data Mining Data Dictionary Views

| View Name | Description |
|-----------------------------|---|
| ALL_MINING_MODELS | Provides information about all accessible mining models |
| ALL_MINING_MODEL_ATTRIBUTES | Provides information about the attributes of all accessible mining models |
| ALL_MINING_MODEL_SETTINGS | Provides information about the settings of all accessible mining models |
| ALL_MINING_MODEL_PARTITIONS | Provides all the model partitions accessible to the user. |
| ALL_MINING_MODEL_VIEWS | Provides a description of the user's own model views. Its columns, except for OWNER, are the same as those in ALL_MINING_MODEL_VIEWS. |
| ALL_MINING_MODEL_XFORMS | Provides the user-specified transformations embedded in all models accessible to the user. |

Related Topics

- *Oracle Database Reference*

Resources For Learning About Oracle Data Miner

This section lists the resources such as documentation, forums, blogs, trainings, and tutorials for Oracle Data Miner.

- Oracle Data Miner Documentation
 - *Oracle Data Miner User's Guide*
 - *Oracle Data Miner Release Notes*
 - *Oracle Data Mining Online Help*
- Oracle Data Mining 12.2 Documentation

- *Oracle Data Mining Concepts*
 - *Oracle Data Mining User's Guide*
- Oracle Data Mining 12.1 Documentation
 - *Oracle Data Mining Concepts*
 - *Oracle Data Mining User's Guide*
- Tutorials
 - [Oracle Data Mining 4.1 OBE \(Oracle By Example\)](#)
 - [Oracle Data Mining 4.0 OBE \(Oracle By Example\)](#)
 - [Text Mining in Oracle Data Miner 4.1](#)
 - [Star Schema Using Oracle Data Miner 4.1](#)
- [Oracle Data Mining Forum](#)
- [Oracle Data Mining Blog](#)
- [Oracle Technology Network - Data Mining](#)

2

Installing the Database to Support Oracle Data Miner

This chapter explains how to install and configure Oracle Database to support Oracle Data Miner.

- [Database Requirements for Oracle Data Miner](#)
Oracle Data Miner 18.3 and later requires Oracle Database Enterprise Edition or Personal Edition 11.2.0.4 or later.
- [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.
- [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.
- [Storage Configuration for Oracle Data Miner](#)
The storage format used by Oracle Data Miner depends on the database version.
- [Installing Oracle Database 11.2.0.4 or Later](#)
To install Oracle Database, follow the installation instructions for your platform and all additional instructions.
- [Connecting to an Oracle Cloud Database Schema](#)
You can connect to an Oracle Cloud Database schema from SQL Developer interface.

Database Requirements for Oracle Data Miner

Oracle Data Miner 18.3 and later requires Oracle Database Enterprise Edition or Personal Edition 11.2.0.4 or later.

To verify that the database meets this requirement, 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 18.3 and later require Oracle Database 12.2 and later.

[Table 2-1](#) lists the Database installation and configuration requirements for Oracle Data Miner.

Table 2-1 Database Requirements Checklist for Oracle Data Miner

| Feature | Requirement | Links to More Information |
|---------------------------|--|--|
| Database version | 11.2.0.4 or later. | Installing Oracle Database 11.2.0.4 or Later |
| Oracle Advanced Analytics | Required for Oracle Data Miner installation. | Oracle Data Miner and Oracle Advanced Analytics |
| Oracle Text | Required for Oracle Data Miner installation. | Oracle Text and Oracle Data Miner |
| XML DB | Required for Oracle Data Miner installation. | XML DB and Oracle Data Miner |
| Storage Configuration | Must be configured for either binary XML storage or object-relational storage, depending on the version of the database. | Storage Configuration for Oracle Data Miner |
| AL32UTF8 Character Set | Required to support multibyte character data and other data formats Oracle highly recommends that you configure the database to support AL32UTF8. | Oracle Database 18c: Oracle Database Globalization Support Guide Oracle Database 12.2: Oracle Database Globalization Support Guide Oracle Database 12.1: Oracle Database Globalization Support Guide |
| Oracle R Enterprise | Required for execution of embedded R scripts in the Oracle Data Miner SQL Query node. | Oracle R Enterprise Installation and Administration Guide |
| Oracle Database Examples | Required for Oracle Data Miner sample data. May be required for Oracle Text Knowledge Base. | Oracle Database 18c: Oracle Database Examples Installation Guide Oracle Database 12.2: Oracle Database Examples Installation Guide |
| Sample Schemas | Required for some Oracle Data Miner sample data. | Oracle Database 12.2: The SH sample schema is not shipped with Oracle Database 12.2. To install the sample schema, go to Oracle Database Sample Schemas Oracle Database 12.1: Oracle Sample Database Sample Schemas |

Related Topics

- [Database Features Used by Oracle Data Miner](#)
Oracle Data Miner uses a number of Oracle Database features such as Oracle Data Mining, Oracle XML DB, Oracle R Enterprise and so on.

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.

Related Topics

- *Oracle Text Reference*
- *Oracle Database Examples Installation Guide*

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.

Related Topics

- *Oracle XML DB Developer's Guide*

Storage Configuration for Oracle Data Miner

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

With Oracle Database 11.2.0.4 and later, 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;
```

Related Topics

- *Oracle Database Administrator's Guide*

Installing Oracle Database 11.2.0.4 or Later

To install Oracle Database, follow the installation instructions for your platform and all additional instructions.

Installation instructions and all additional instructions are specified in the topic *Database Requirements for Oracle Data Miner*.

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

- Linux:
 - Oracle Database 18c: [Oracle Database Installation Guide for Linux](#)
 - Oracle Database 12.2: [Oracle Database Installation Guide for Linux](#)
 - Oracle Database 12.1: [Oracle Database Installation Guide for Linux](#)
 - Oracle Database 11.2.0.4: [Oracle Database Installation Guide for Linux](#)
- Oracle Solaris:
 - Oracle Database 18c: [Oracle Database Installation Guide for Oracle Solaris](#)
 - Oracle Database 12.2: [Oracle Database Installation Guide for 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 12.2: [Oracle Database Installation Guide for Microsoft Windows](#)
 - Oracle Database 12.1: [Oracle Database Installation Guide for Microsoft Windows](#)
 - Oracle Database 11.2.0.4: [Oracle Database Installation Guide for Microsoft Windows](#)

To install Oracle Database on other platforms, search the Oracle Help Center for your platform-specific installation instructions.

Related Topics

- [Oracle Help Center](#)
- [Database Requirements for Oracle Data Miner](#)
Oracle Data Miner 18.3 and later requires Oracle Database Enterprise Edition or Personal Edition 11.2.0.4 or later.

Connecting to an Oracle Cloud Database Schema

You can connect to an Oracle Cloud Database schema from SQL Developer interface.

From the SQL Developer interface, you can create or edit a connection to an Oracle Cloud Database instance, which does not offer Oracle Net (SQL*Net) access. Oracle Data Miner supports Oracle Database Cloud Service (Database as a Service), and Oracle Database Exadata Cloud Service.

See Also:

For more information about how to create or edit a connection to an Oracle Cloud Database Schema Service instance, see *Oracle SQL Developer User's Guide*.

Related Topics

- [Oracle SQL Developer User's Guide](#)

3

Installing Oracle Data Miner

This chapter explains how to install SQL Developer and the Oracle Data Miner repository.

- [Oracle Data Miner Installation Overview](#)
Oracle Data Miner installation refers to installation of the Oracle Data Miner repository in an Oracle Database. The repository serves as application manager and workflow storage manager for Oracle Data Miner.
- [About Oracle Data Miner Sample Data](#)
Oracle Data Miner contains sample data that includes tables and views.
- [Downloading SQL Developer](#)
Oracle SQL Developer is available for download free of charge from the Oracle Technology Network.
- [Installing the Repository Using a Script](#)
You can install the Oracle Data Miner repository by running the `installodmr` script.
- [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.
- [Installing the Repository Using SQL Developer GUI](#)
You can install the Oracle Data Miner repository using the SQL Developer graphical user interface.
- [Installing JSON Parser and Data Guide](#)
JSON query processing is available in Oracle Database 12.1.0.2 and later.

Oracle Data Miner Installation Overview

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

To install Oracle Data Miner:

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

Related Topics

- [Downloading SQL Developer](#)
Oracle SQL Developer is available for download free of charge from the Oracle Technology Network.
- [Installing the Database to Support Oracle Data Miner](#)
- [Oracle Data Miner User's Guide](#)
- [Database Requirements for Oracle Data Miner](#)
Oracle Data Miner 18.3 and later requires Oracle Database Enterprise Edition or Personal Edition 11.2.0.4 or later.

About Oracle Data Miner Sample Data

Oracle Data Miner contains sample data that includes tables and views.

Note:

The SH sample schema is not shipped with Oracle Database 12.2. To install the sample schema, go to DB Sample Schemas.

The sample tables and views are:

- `MINING_DATA_BUILD_V`, `MINING_DATA_TEST_V`, `MINING_DATA_APPLU_V`: These are views based on tables in the SH schema. This data is used in Oracle Data Mining sample programs, as described in Oracle Data Mining Application Developer's Guide and Oracle Data Mining Administrator's Guide.
- `MINING_DATA_TEXT_BUILD_V`, `MINING_DATA_TEXT_TEST_V`, `MINING_DATA_TEXT_APPLU_V`: These are 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.
- `ODMR_CARS_DATA`: This is sample data about US automobiles, for experimenting with the Graph node.
- `INSUR_CUST_LVT_SAMPLE`: This is sample data used by the Oracle By Example (OBE) tutorials for Data Mining.
- `ODMR_SALES_JSON_DATA`: This is sample sales data for experimenting with the JSON query node in Oracle Database 12.1.0.2 and later.
- `ODMR_MINING_DATA_TEXT`, `ODMR_SALES_DATA`: This is sample data that can be used as a partial replacement, if SH schema is not easily available. It is loaded when Oracle Data Miner demo data is installed, without SH dependencies.
- `WIKISAMPLE`: This is sample data to demonstrate the Explicit Feature Extraction node.

Related Topics

- [Oracle Database Sample Schemas.](#)

Downloading SQL Developer

Oracle SQL Developer is available for download free of charge from 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:

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 later. 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.

Related Topics

- [SQL Developer Download page.](#)
- [SQL Developer Documentation Index.](#)

Installing the Repository Using a Script

You can install the Oracle Data Miner repository by running the `installodmr` script.

Before you install the repository, verify disk space availability. You must have 4 MB disk space for initial storage allocation. For additional storage, you must have an average of 1 MB disk space to handle 10 workflows of 20 nodes each.

 **Note:**

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

To install the Oracle Data Miner repository:

1. Log in to the database with DBA role.
2. Run the `installodmr` script.

```
installodmr.sql default_tablespace temp_tablespace
```

For example, if you have set the default search path, 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, 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 later), 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.

Related Topics

- [Setting the Path to Oracle Data Miner Scripts](#)
You can set the path to Oracle Data Miner scripts using SQL*Plus or SQL Developer Worksheet.
- [Storage Configuration for Oracle Data Miner](#)
The storage format used by Oracle Data Miner depends on the database version.
- [Granting or Dropping Access Rights to 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 also revoke access rights by running a script.
- [About Oracle Data Miner Administration Scripts](#)
Oracle Data Miner includes a set of Structured Query Language (SQL) scripts for installing and managing the repository.

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. If the `SH` schema is not present in the database, then the script prepares only the demo data that does not depend on `SH`.

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

1. Log in to the database with DBA role.
2. Verify that the `SH` schema is present in the database.
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 with DBA role.
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
```

Related Topics

- [Installing the Repository Using a Script](#)
- [Setting the Path to Oracle Data Miner Scripts](#)
- [About Oracle Data Miner Sample Data](#)

Installing the Repository Using SQL Developer GUI

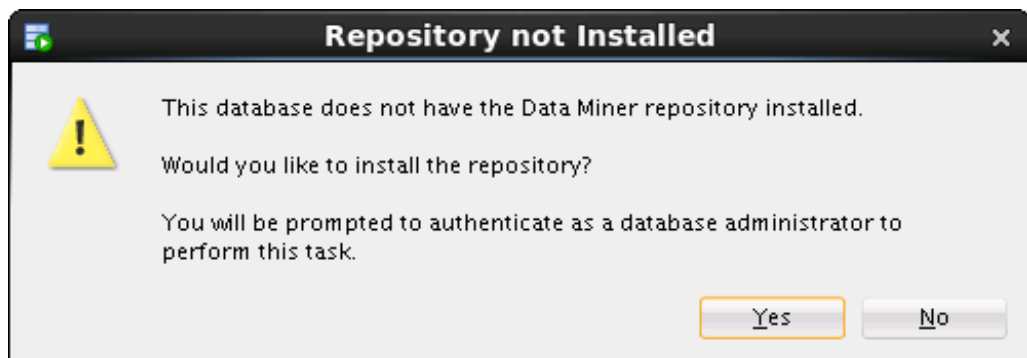
You can install the Oracle Data Miner repository using the SQL Developer graphical user interface.

Before you install the repository, verify disk space availability. You must have 4 MB disk space for initial storage allocation. For additional storage, you must have an average of 1 MB disk space to handle 10 workflows of 20 nodes each.

To install the Oracle Data Miner repository from SQL Developer GUI:

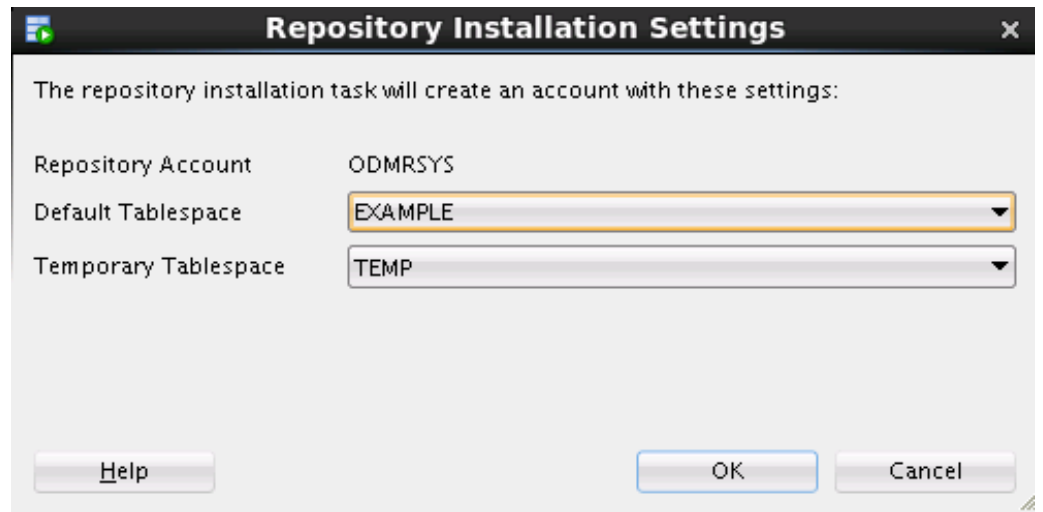
1. In the **Connections** tab of SQL Developer, select an administrative connection to the target database, or create a new one.
2. Create an Oracle Data Miner user:
 - a. Drill on the administrative connection.
 - b. Right click **Other Users** and choose **Create User**.
 - c. Grant the `CONNECT` role to the user, and for **Default Tablespace**, specify **Unlimited Quota**.
3. In the **Connections** tab, create a connection for the Oracle Data Miner user.
4. From the **View** menu, select **Data Miner**, and 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:

Figure 3-1 Repository not Installed



9. Click **Yes** to install the repository.
10. After you provide the password, information about the repository is displayed. For example:

Figure 3-2 Repository Installation Settings



11. Click **OK**.
12. The Repository Installation Settings 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 sample data is installed. If SH is not installed, then only ODMR_CARS_DATA, INSUR_CUST_LVT_SAMPLE, ODMR_SALES_JSON_DATA ODMR_MINING_DATA_TEXT, ODMR_SALES_DATA, and WIKISAMPLE (Database 12.1 only) are installed.

 **Note:**

The SH sample schema is not shipped with Oracle Database 12.2. To install the sample schema, go to DB Sample Schemas.

Figure 3-3 Install Data Miner Repository



 **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.

13. When the installation is complete, click **Close**.

Related Topics

- [About Oracle Data Miner Sample Data](#)
Oracle Data Miner contains sample data that includes tables and views.
- [Oracle Database Sample Schemas](#).
- [Data Mining APIs](#).

Installing JSON Parser and Data Guide

JSON query processing is available in Oracle Database 12.1.0.2 and later.

In Oracle Database 12.2 and later, you do not have to install the JSON parser as Oracle Data Miner uses the JSON parser provided by the database.

 **Note:**

If you have installed or migrated the repository using scripts, then separate postinstallation step to enable JSON support in Oracle Data Miner is not required.

 **Note:**

You must have JVM installed in the Oracle Database.

To verify if Oracle JVM is installed in the Oracle Database, run this query:

```
select * from registry$ where cid like '%JAVA%'
```

- [About the Scripts for Installing JSON Support](#)
To install JSON parser and the schema generator for Oracle Data Miner, two scripts are available.
- [Requirements for Running the Scripts to Install JSON Parser](#)
The scripts `loadjsonschemagen.sql` and `loadjsonschemagenWithSN.sql` install the JSON parser and schema generator for Oracle Data Miner.
- [Installing JSON Parser in a Remote Database](#)
You can install JSON support in a remote database by running the `loadjsonschemagen` or `loadjsonschemagenWithSN` scripts.

- [Installing JSON Parser Locally on the Database Host](#)
You can install JSON parser locally on a database host by using the scripts `loadjsonschemagen.sql`, `validateODMRSYS.sql`, `org.glassfish.javax.json.jar`, and `JSONSchemaGenerator.jar`.

Related Topics

- [Configuring Oracle JVM](#)

About the Scripts for Installing JSON Support

To install JSON parser and the schema generator for Oracle Data Miner, two scripts are available.

The only difference between the two scripts in their arguments is that one accepts the Oracle SID, the other accepts the Oracle Service Name. You can choose the script that best suits your database installation. If you are using a pluggable database, then the Oracle Service Name is required.

Note:

You must use the SYS account to run these scripts.

- `loadjsonschemagen.sql` uses the Oracle SID.

```
loadjsonschemagen.sql sysuser syspassword host port SID jar_path
```

- `loadjsonschemagenWithSN.sql` uses the Oracle Service Name.

```
loadjsonschemagenWithSN.sql sysuser syspassword host port service_name  
jar_path
```

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

```
org.glassfish.javax.json.jar
```

```
JSONSchemagenerator.jar
```

Requirements for Running the Scripts to Install JSON Parser

The scripts `loadjsonschemagen.sql` and `loadjsonschemagenWithSN.sql` install the JSON parser and schema generator for Oracle Data Miner.

To run the scripts, ensure the following requirements:

- 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.

- Java must be installed in Oracle Database. To verify JVM installation in the database, run this query:

```
select * from registry$ where cid like '%JAVA%'
```

Sample output:

```
JAVAVM JServer JAVA Virtual Machine 0 0 12.1.0.2.0  
CATJAVA Oracle Database Java Packages 0 0 12.1.0.2.0
```

Installing JSON Parser in a Remote Database

You can install JSON support in a remote database by running the `loadjsonschemagen` or `loadjsonschemagenWithSN` scripts.

1. Log in to the computer where SQL Developer is installed.
2. Navigate to the `scripts` directory:

```
cd SQL_Developer_Home\sqldeveloper\dataminer\scripts
```

3. Start SQL*Plus with DBA privilege.

```
SQLPLUS / as sysdba
```

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

```
@loadjsonschemagen user password host port SID jar_path  
or
```

```
@loadjsonschemagenWithSN user password host port service_name jar_path
```

where the value of `jar_path` is:

```
c:\SQL_Developer_home\sqldeveloper\extensions  
\oracle.dmt.dataminer\lib
```

Installing JSON Parser Locally on the Database Host

You can install JSON parser locally on a database host by using the scripts `loadjsonschemagen.sql`, `validateODMRSYS.sql`, `org.glassfish.javax.json.jar`, and `JSONSchemaGenerator.jar`.

To install JSON parser locally on a database host:

1. Copy the scripts `loadjsonschemagen.sql` and `validate ODMRSYS.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 with DBA privilege.

SQLPLUS / as sysdba

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

Related Topics

- [About the Scripts for Installing JSON Support](#)

4

Managing Oracle Data Miner Users

This chapter explains how to manage Oracle Data Miner user accounts.

- [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.
- [About Proxy Users for Oracle Data Miner](#)
SQL Developer provides support for proxy users that have their own login credentials but share the same target database user account.
- [Setting the JVM Property](#)
Oracle SQL Developer LDAP connections utilizing a proxy or target user protocol, can generate an SQL run time error when running the SQL Query Node.
- [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.
- [Granting or Dropping Access Rights to 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 also revoke access rights by running a script.

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`.

- [Controlling the Size of Users' Schemas](#)
You must control the size of users' schema to ensure conservation of storage in the schema.
- [Objects in Oracle Data Miner Users' Schema](#)
Workflows create objects, tables and views. Oracle Data Miner stores these objects in the user's schema.
- [About Oracle Data Miner Internal Tables](#)
Internal tables in the user's schema store information that supports workflows and data mining activities.

Related Topics

- [Oracle Data Miner Architecture](#)
Oracle Data Miner is an extension of Oracle SQL Developer, a graphical development environment for Oracle SQL.
- [About 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.

Controlling the Size of Users' Schemas

You must control the size of users' schema to ensure conservation of storage in the schema.

To conserve storage in users' schema:

- Delete workflows when they are no longer needed.
- Export workflows, and then delete them in the user's schema.
- Use a separate tablespace for ODMRSYS to isolate repository storage consumption from user tablespaces.

Related Topics

- [Managing the Oracle Data Miner Repository](#)

Objects in Oracle Data Miner Users' Schema

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

In the user's schema:

- Models, tables, and views that are directly named by the user through the Data Miner node editors. These include mining models created by the Model node and tables created by the Create Table node.
- Tables and views that are created by Data Miner nodes to store and view generated results, but are not directly named by users through the Data Miner node editors. 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.

About Oracle Data Miner Internal Tables

Internal tables in the user's schema store information that supports workflows and data mining activities.

The internal tables perform the following:

- 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 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.

About Proxy Users for Oracle Data Miner

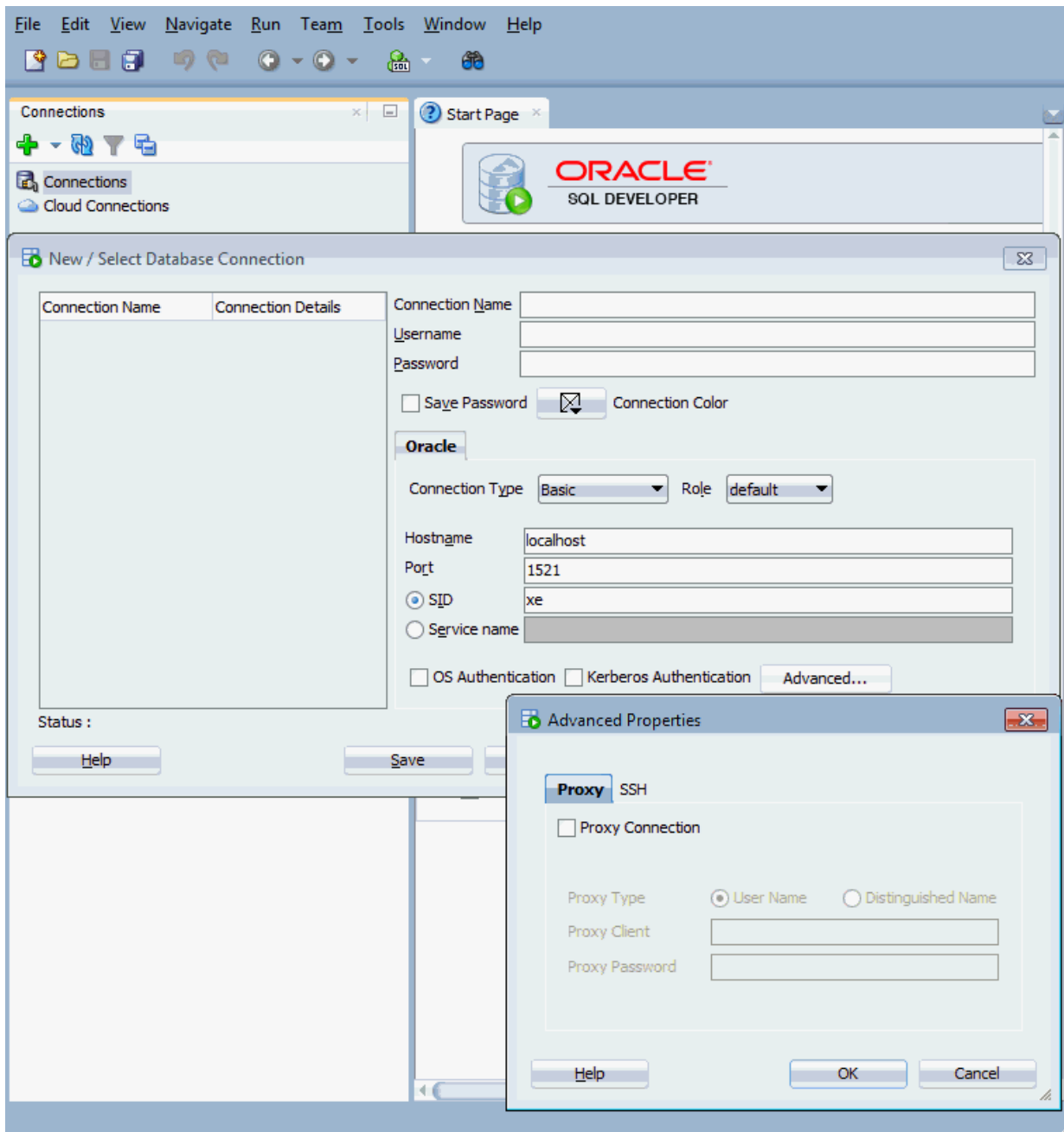
SQL Developer provides support for proxy users that have their own login credentials but share the same target database user account.

A SQL Developer connection typically provides database access to a single user that is defined in that database. SQL Developer has several connection types that support the creation of proxy users.

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.

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.

Figure 4-1 Proxy User for a Basic Connection in SQL Developer



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.

Related Topics

- Database Connections
- Connections with Proxy Authentication
- Create and Edit Database Connections

- [Configuring Authentication](#)

Setting the JVM Property

Oracle SQL Developer LDAP connections utilizing a proxy or target user protocol, can generate an SQL run time error when running the SQL Query Node.

If you encounter the invalid name pattern error when running a query as a proxy user, then set the JVM property

```
oracle.jdbc.createDescriptorUseCurrentSchemaForSchemaName to true.
```

To set the JVM property, open Oracle SQL Developer and run the following from the command prompt: `C:\<SQL Developer Installation Directory>\sqldeveloper\ide\bin\sqldeveloper.exe -`

```
JDoracle.jdbc.createDescriptorUseCurrentSchemaForSchemaName=true
```

To make a permanent change, update the `sqldeveloper.conf` file and include the property definition, as follows:

1. Shut down Oracle SQL Developer.
2. Open the `sqldeveloper.conf` file and update it with the property definition:

```
C:\<SQL Developer Installation Directory>\sqldeveloper\ide\bin  
\sqldeveloper.exe -  
JDoracle.jdbc.createDescriptorUseCurrentSchemaForSchemaName=true
```

3. Save the `sqldeveloper.conf` file to `<SQL Developer Installation Directory>\sqldeveloper\bin\sqldeveloper.conf`. This overwrites the existing file.

With the updated `sqldeveloper.conf` file, you do not have to start Oracle SQL Developer from the command line.

The top of the `sqldeveloper.conf` file is shown below with the new text in bold:

```
IncludeConfFile ../../ide/bin/ide.conf
```

```
SetJavaHome ../../jdk
```

```
#Workaround for LDAP Proxy failure
```

```
AddVMOption
```

```
-Doracle.jdbc.createDescriptorUseCurrentSchemaForSchemaName=true
```

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.

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.

- [Single User Access](#)
In Single User Access, 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.
- [Multiple User Access](#)
In Multiple User Access, there are multiple user schemas. A schema can support an individual user, or it can support a workgroup of proxy users.
- [Shared User Environment](#)
Shared user environments facilitate collaboration. Oracle Data Miner uses locking mechanisms to coordinate access to workflows, when several proxy users share access to a single database account.
- [About the Document in Use Condition](#)
The `Document in Use` message is generated when a user tries to edit a workflow while the workflow sessions are running in the database.

Single User Access

In Single User Access, 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. Single user access ensures private workspaces but does not promote collaboration.

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.

Multiple User Access

In Multiple User Access, there are multiple user schemas. A schema can support an individual user, or it can support a workgroup of proxy users.

You can set up some combination of individual and shared access. You can also use proxy authentication for all users, even for unshared users.

Shared User Environment

Shared user environments facilitate collaboration. Oracle Data Miner uses locking mechanisms to coordinate access to workflows, when several proxy users share access to a single database account.

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.

About the Document in Use Condition

The `Document in Use` message is generated when a user tries to edit a workflow while the workflow sessions are running in the database.

If an Oracle Data Miner client disconnects from the network (for example, if a cable is disconnected or a laptop goes into deep sleep), then the 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, the *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 the topic "Terminating Sessions" in *Oracle Database Administrator's Guide* for instructions.

Related Topics

- [Oracle Database Administrator's Guide](#)

Granting or Dropping Access Rights to 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 also revoke access rights by running a script.

- [Granting Access Rights Using the GUI](#)
When you install Oracle Data Miner repository using the Graphical User Interface (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.
- [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.
- [Dropping Access Rights Using a Script](#)
You can drop access rights to the Oracle Data Miner repository by executing the `dropusergrants` script.
- [Granting Access to Data](#)
You must have read access or the `SELECT` permission to data that is used for building mining models or for scoring.

Granting Access Rights Using the GUI

When you install Oracle Data Miner repository using the Graphical User Interface (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.

Related Topics

- [About Oracle Data Miner Sample Data](#)
Oracle Data Miner contains sample data that includes tables and views.
- [Installing the Repository Using SQL Developer GUI](#)
You can install the Oracle Data Miner repository using the SQL Developer graphical user interface.

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.

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.

Related Topics

- [Dropping Access Rights Using a Script](#)
You can drop access rights to the Oracle Data Miner repository by executing the `dropusergrants` script.

Granting Access to Data

You must have read access or the `SELECT` permission to data that is used for building mining models or for scoring.

You must grant `SELECT` permission directly to a target user. Do not grant 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.

5

Managing the Oracle Data Miner Repository

This chapter explains how to use scripts to manage the Oracle Data Miner repository.

- [About Oracle Data Miner Administration Scripts](#)
Oracle Data Miner includes a set of Structured Query Language (SQL) scripts for installing and managing the repository.
- [Setting the Path to Oracle Data Miner Scripts](#)
You can set the path to Oracle Data Miner scripts using SQL*Plus or SQL Developer Worksheet.
- [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.
- [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.
- [Migrating the Repository](#)
This section contains topics related to repository migration.
- [Dropping the Repository](#)
The `dropRepositoryAndUserObjects` script drops the Oracle Data Miner repository and related objects in the users' schemas.

About Oracle Data Miner Administration Scripts

Oracle Data Miner includes a set of Structured Query Language (SQL) scripts for installing and managing the repository.

The SQL scripts are installed with SQL Developer in the following directory:

```
SQL_Developer_Home\sqldeveloper\dataminer\scripts
```

You can run the SQL scripts in SQL*Plus or in SQL Developer Worksheet. All the Oracle Data Miner scripts must be run with DBA privilege.

 **Note:**

Many of the Oracle Data Miner scripts are integrated with SQL Developer, enabling access to some administrative functions through the Data Graphical User Interface.

Setting the Path to Oracle Data Miner Scripts

You can set the path to Oracle Data Miner scripts using SQL*Plus or SQL Developer Worksheet.

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 must change the maximum number of rows to print in a script to 500000.

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:

```
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');
PROPERTY_NAME          PROPERTY_STR_VALUE
-----
REPOSITORY_STATUS      LOADED
VERSION                12.1.0.2.3
WF_VERSION             12.1.0.2.3
```

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`.

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.

- **Full Backup and Restore**
For Oracle Databases 11.2.0.4 and later, you can perform a full backup and restore of the Oracle Data Miner repository and user schema independently of a full database backup.

- [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.
- [Workflow Only Restore](#)
You can restore workflows from the backup table created by `createxmlworkflowsbackup2`.
- [Workflow Only Restore Examples](#)
This topic provides examples on selective workflow restore and full workflow restore.

Related Topics

- [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 Architecture](#)
Oracle Data Miner is an extension of Oracle SQL Developer, a graphical development environment for Oracle SQL.

Full Backup and Restore

For Oracle Databases 11.2.0.4 and later, you can perform a full backup and restore of the Oracle Data Miner repository and user schema independently of a full database backup.

- [Full Backup and Restore in Database 11.2.0.4 and Later](#)
In Oracle Database 11.2.0.4 and later, the XML storage in the Oracle Data Miner repository is binary. In these databases, you can use Oracle Data Pump to back up and restore `ODMRSYS` and the user schemas independently of a full backup and restore of the database.

Full Backup and Restore in Database 11.2.0.4 and Later

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

Using Oracle Data Pump, you can back up and restore individual schemas. Alternatively you can back up and restore Oracle Data Miner with Oracle RMAN.

Related Topics

- [Oracle Database Utilities](#)

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.

The simplified backup strategy safeguards the workflow specifications and enables you to restore a workflow if you accidentally delete it. After the workflow is restored,

you must 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`.

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

```
set serveroutput on
@createxmlworkflowsbackup21 BACKUPACCT
```

**Note:**

The `dropRepositoryAndUserObjects` script drops all the backup tables when it drops the repository. If you run the `dropRepositoryAndUserObjects` script to drop the repository, then all the workflow backups are lost.

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.

Related Topics

- [Dropping the Repository](#)
The `dropRepositoryAndUserObjects` script drops the Oracle Data Miner repository and related objects in the users' schemas.

Workflow Only Restore

You can restore workflows from the backup table created by `createxmlworkflowsbackup2`.

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:

```
restorexmlworkflowfrombackup2.sql [option] [backup_account]  
[workflow_definition]
```

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 DROP_AND_ADD BACKUPACCT
```

Related Topics

- [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.

Workflow Only Restore Examples

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

Example 5-1 Selective Workflow Restore

Let us 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 with DBA privilege, 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 us 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
```

Migrating the Repository

This section contains topics related to repository migration.

Topics include:

- [Upgrading 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.
- [Scripts to Migrate the Repository](#)
Starting with SQL Developer 4.0, Oracle Data Miner migration scripts are available for specific upgrade paths.
- [Upgrading 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.
- [Upgrading 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.

Upgrading 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
```

Scripts to Migrate the Repository

Starting with SQL Developer 4.0, Oracle Data Miner migration scripts are available for specific upgrade paths.

Table 5-1 Oracle Data Miner Upgrade Scripts

| SQL Developer Version | Database Version | Script | Description |
|-----------------------|---|---|---|
| 4.0 and later | 11.2.0.4 and later | <code>migratebinaryodmr.sql</code> | Relevant for future releases, when XML conversion from object storage to binary storage will no longer be needed. |
| 4.0 and later | 11.2.0.3 or earlier originally, and then the database was upgraded to 11.2.0.4 or later | <code>upgradeRepoWithNewTableSpace.sql</code> or <code>upgradeRepoFromORtoBinary.sql</code> | 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 <code>ODMRSYS</code> tablespace is ASSM or not. |

Table 5-1 (Cont.) Oracle Data Miner Upgrade Scripts

| SQL Developer Version | Database Version | Script | Description |
|-----------------------|--|----------------------------------|--|
| 3.2.2 or earlier | 11.2.0.4 and later, where the default ODMRSYS tablespace is not ASSM-based | upgradeRepoWithNewTableSpace.sql | ASSM tablespace required as input parameter. This will be used to migrate the workflow data from XML object storage to XML binary storage. |
| 3.2.2 or earlier | 11.2.0.4 and later, where the default ODMRSYS tablespace is ASSM-based | upgradeRepoFromORtoBinary.sql | Workflow data will be migrated from XML object storage to XML binary storage. |

Upgrading 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
```

Related Topics

- *Oracle Database Administrator's Guide*

Upgrading 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
```

Related Topics

- *Oracle Automatic Storage Management Administrator's Guide*

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
```

Related Topics

- [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 Architecture](#)
Oracle Data Miner is an extension of Oracle SQL Developer, a graphical development environment for Oracle SQL.
- [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.

6

Managing System Resources for Oracle Data Miner

This chapter provides information to help you optimize your system to support Oracle Data Miner.

- [Oracle Data Miner Resource Management Overview](#)
You can effectively manage system resources for Oracle Data Miner by using features of Oracle Database and the options provided in Oracle Data Miner repository.
- [Allocating Resources to Oracle Data Miner User Sessions](#)
You can use Database Resource Manager to create resource plans that allocate system resources for groups of sessions based on session attributes.
- [Managing Model Builds](#)
The process of building mining models can consume significant system resources. You have the option to control the impact of model builds on overall system resources, by increasing or decreasing the value of `MAX_NUM_THREADS` repository property.
- [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.
- [Managing Parallel Processing](#)
Oracle Data Miner workflows and views, and most Data Mining algorithms, can take advantage of parallel processing of queries when it is enabled in the database. Parameters in `INIT.ORA` control the behavior of parallel processing. By default parallelism is disabled (`PARALLEL_DEGREE_POLICY=MANUAL`).
- [Summary of Oracle Data Miner Repository Properties for System Management](#)
A summary of the system management properties is available in the Oracle Data Miner repository.

Oracle Data Miner Resource Management Overview

You can effectively manage system resources for Oracle Data Miner by using features of Oracle Database and the options provided in Oracle Data Miner repository.

You can effectively manage system resources for Oracle Data Miner in the following ways:

- 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 used for 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 query processing, change the Oracle Data Miner default parallel query setting to prevent users from specifying parallel processing for individual nodes and workflows.

Allocating Resources to Oracle Data Miner User Sessions

You can use Database Resource Manager to create resource plans that allocate system resources for groups of sessions based on session attributes.

Oracle Database Resource Manager allocates CPU time, configures parallel query processing, limits the number of sessions, and controls other aspects of system behavior that would otherwise be controlled by the operating system. 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.

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_pl => 50,
    utilization_limit => 55,
    parallel_degree_limit_pl => 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_pl => 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';
-- DBA can also enable a resource plan for the period of time
corresponding
-- to a job scheduler Window (via the dbms_scheduler.create_window
procedure)

```

Related Topics

- *Oracle Database Administrator's Guide*

Managing Model Builds

The process of building mining models can consume significant system resources. You have the option to control the impact of model builds on overall system resources, by increasing or decreasing the value of `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.

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
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

```

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';

```

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';
```

Related Topics

- *Oracle Database Administrator's Guide*
- *Oracle Database PL/SQL Packages and Types Reference*

Managing Parallel Processing

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

Parallel processing of queries can be system-determined, or it can be set to Degree of Parallelism. When parallel processing 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 processing 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 processing feature of Oracle Data Miner.

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.

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';

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';
```

Related Topics

- *Oracle Database VLDB and Partitioning Guide*

Summary of Oracle Data Miner Repository Properties for System Management

A summary of the system management properties is available in the Oracle Data Miner repository.

[Table 6-1](#) provides a summary of the system management properties available in the Oracle Data Miner repository.

Table 6-1 Oracle Data Miner Repository Properties for System Management

| Property | Type | Description |
|---------------------------|------------------------|---|
| PARALLEL_QUERY_ON_ALLOWED | Boolean | Indicates whether users can specify parallel query for nodes or workflows. Values are TRUE or FALSE. Default is TRUE. |
| MAX_NUM_THREADS | Integer | Maximum number of concurrent model builds. Default is 10. |
| THREAD_WAIT_TIME | Integer | 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. |
| MAX_THREAD_WAIT | Integer | 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. |
| MAX_STRING_SIZE | VARCHAR2, NVARCHAR2 | Controls the maximum size of the data types VARCHAR2 and NVARCHAR2 in tables that are generated by Oracle Data Miner. This property is initialized from the value of MAX_STRING_SIZE database property. |
| WORKFLOW_JOB_CLASSES | Varchar | Oracle Scheduler job class for workflows. Default is DEFAULT_JOB_CLASS. |
| POLLING_IDLE_RATE | NUMBER | This is the rate at which Oracle Data Miner client polls the database when there are apparently no workflows detected as Running. Default is 30000 milliseconds (internal default). The actual value is 0 in the repository property table. If the administrator changes its value to a non zero value, then it becomes the active setting. |

 **Note:**

If you change the MAX_STRING_SIZE database property, then you must update the MAX_STRING_SIZE property in the ODMR\$REPOSITORY_PROPERTIES to match the value. ODMr run time error may occur if the values are not in synchrony.

Table 6-1 (Cont.) Oracle Data Miner Repository Properties for System Management

| Property | Type | Description |
|-------------------------------|----------|---|
| POLLING_ACTIVE_RATE | NUMBER | This is the rate at which Oracle Data Miner client polls the database when there are workflows detected as Running. Default is 1500 milliseconds (internal default). The actual default in the repository property setting is 0. If you change it to a non zero value, then it becomes the active setting. |
| POLLING_COMPLETE_WINDOW | NUMBER | This is the window of time in hours to include completed workflows in the polling query result. Default is 0 in the repository. Client level default is 24. This setting will override the client level preference setting available to the user if it is changed to a non zero value. |
| PURGE_WORKFLOW_SCHEDULER_OBJS | VARCHAR2 | <p>Purges old Oracle Scheduler objects generated by the running workflows.</p> <ul style="list-style-type: none"> • TRUE (Default). Purges Oracle Scheduler objects automatically • FALSE. Does not purge any Oracle Scheduler objects. If a workflow is deleted, then Oracle Scheduler objects are purged. <p>Oracle recommends that you use the FALSE option if you must triage workflow issues. You must set it to TRUE after completing the triage.</p> |
| PURGE_WORKFLOW_EVENT_LOG | NUMBER | Controls the number of workflows runs that are preserved for each workflow in the event log. The events of the older workflow are purged to keep within this limit. Default is 2. |

 **Note:**

If you provide the value 0, then Oracle Data Miner will not purge any workflow events. This is not recommended as it will increase the size of the event log, and affect query performances.

7

Generating and Deploying SQL Scripts

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

- [Overview of the SQL Script Generation Feature](#)
Oracle Data Miner provides the SQL script generation feature, using which you can generate SQL scripts for one or all nodes in a workflow. You can then integrate the SQL scripts into another application.
- [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`.
- [Generating SQL Script Files from the Workflow](#)
You must run the workflow before generating the SQL script from it.
- [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.
- [Deploying SQL Scripts on the Target Database](#)
Deploying the SQL scripts involves running the master script file from the base directory.

Overview of the SQL Script Generation Feature

Oracle Data Miner provides the SQL script generation feature, using which 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 details supported by a use case.

Related Topics

- [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`.

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
- [Premise of the SQL Script Generation Use Case](#)
The SQL Script Generation use case described in this section is based on the following premises:
- [About the Sample Workflow](#)
The sample workflow `codegen_workflow`, demonstrates data modelling and data scoring.
- [Performing Prerequisite Tasks](#)
Before deploying a workflow, there are certain tasks which must be performed.
- [Importing and Running a Workflow](#)
You can import a predefined workflow into a project.

Related Topics

- [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_home\dataminer\demos\workflows`.

Premise of the SQL Script Generation Use Case

The SQL Script Generation use case described in this section is based on the following premises:

- 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`.
- [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_home\dataminer\demos\workflows`.

Related Topics

- [About the Sample Workflow](#)
The sample workflow `codegen_workflow`, demonstrates data modelling and data scoring.

- **Performing Prerequisite Tasks**
Before deploying a workflow, there are certain tasks which must be performed.
- **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.

Location of Demo Workflow Files

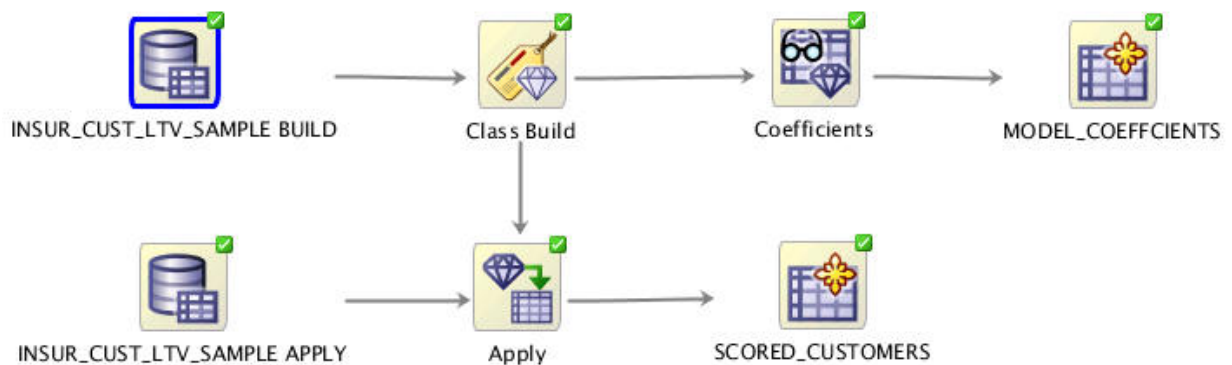
The workflow file `codegen_workflow.xml` which contains the predefined workflow is available in the SQL Developer installation location at: `sqldeveloper_home\dataminer\demos\workflows`.

About the Sample Workflow

The sample workflow `codegen_workflow`, demonstrates data modelling and data scoring.

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.

Figure 7-1 The Sample Workflow - codegen_workflow



- **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.

Related Topics

- [Premise of the SQL Script Generation Use Case](#)
The SQL Script Generation use case described in this section is based on the following premises:
- [Performing Prerequisite Tasks](#)
Before deploying a workflow, there are certain tasks which must be performed.
- [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_home\dataminer\demos\workflows`.

Performing Prerequisite Tasks

Before deploying a workflow, there are certain tasks which must be performed.

The prerequisite tasks are:

1. Install Oracle Data Miner 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.

```
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.

Related Topics

- [Installing Oracle Data Miner](#)
- [Oracle By Example Tutorials](#)
- [Importing and Running a Workflow](#)
You can import a predefined workflow into a project.
- [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.

Importing and Running a Workflow

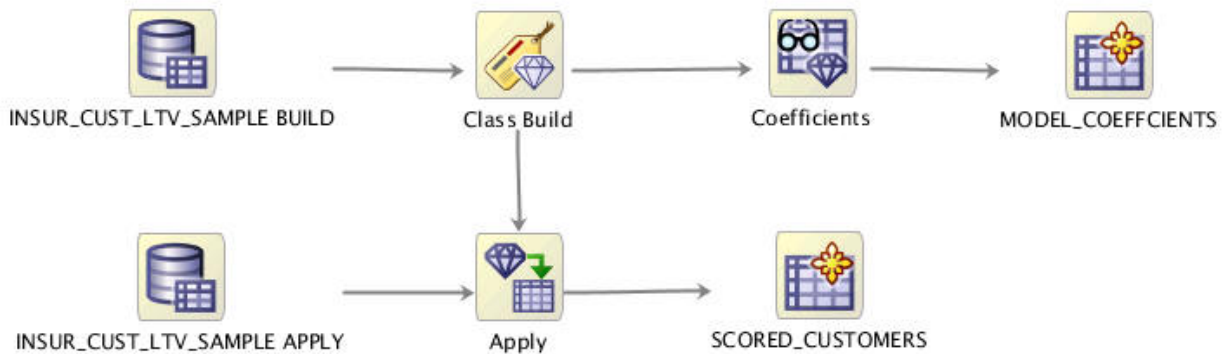
You can import a predefined workflow into a project.

To import the predefined `workflow_codegen.xml`:

1. In SQL Developer, go to the **Data Miner** tab and expand the connection.
2. Right-click the connection and click **New Project**.

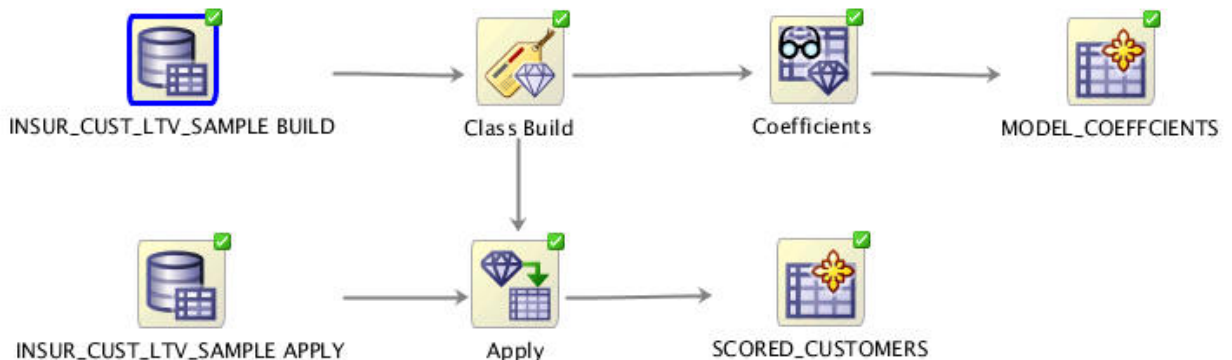
3. Right-click the project that you just created and select **Import Workflow**.
4. In the Import Workflow dialog box, browse to the location where you have downloaded and saved the sample workflow file `codegen_workflow.xml`. Select the `codegen_workflow.xml` and click **OK**. The imported workflow `codegen_workflow` is now displayed in the Oracle Data Miner UI, as shown in [Figure 7-2](#).

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](#).

Figure 7-3 The `codegen_workflow` after run



This completes the task of importing and running the workflow.

Related Topics

- [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.
- [Deploying SQL Scripts on the Target Database](#)
Deploying the SQL scripts involves running the master script file from the base directory.
- [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_home\dataminer\demos\workflows`.

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`
 - `INSUR_CUST_LTV_SAMPLE APPLY`
 - `Apply`
 - `SCORED_CUSTOMERS`

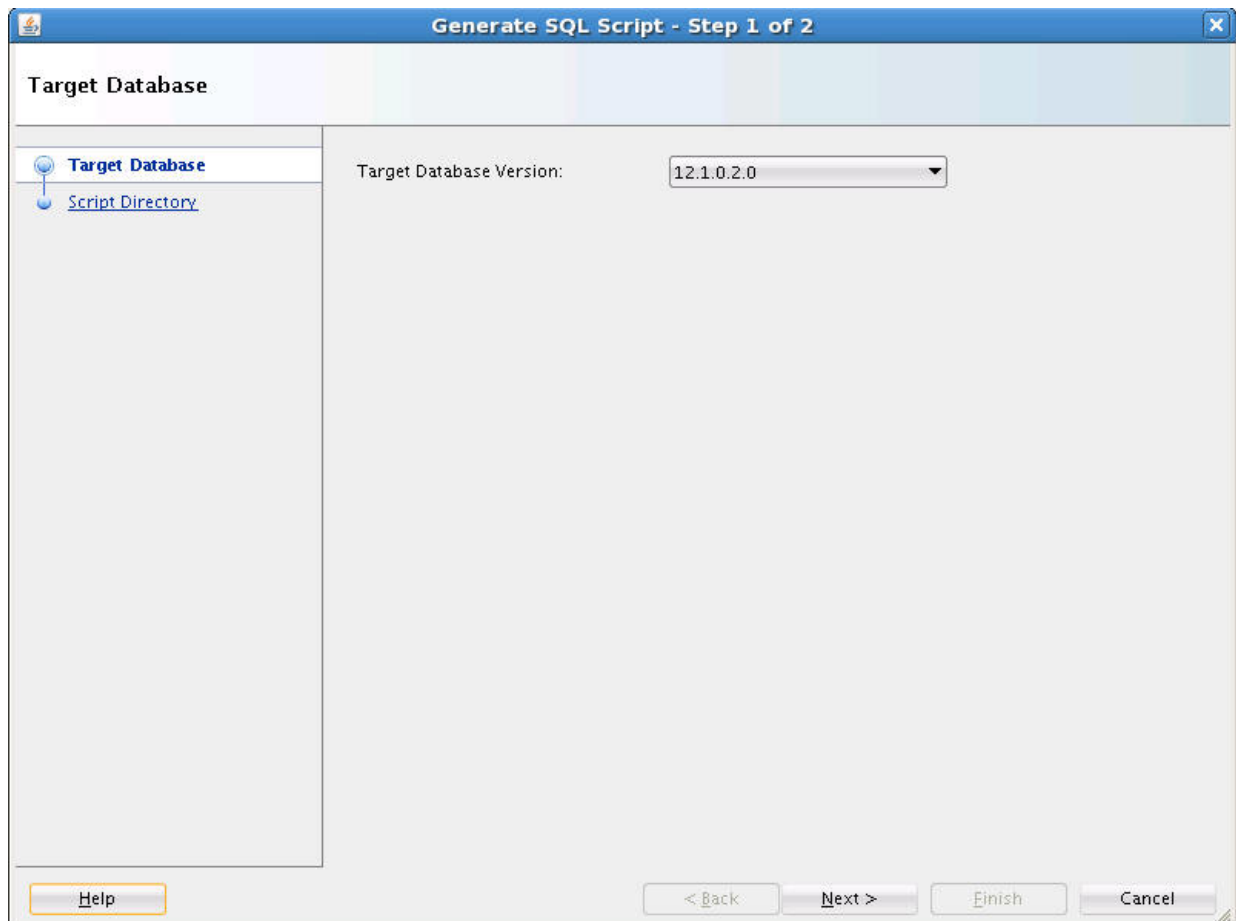
- **Selected Node and Connected Nodes:** This option generates scripts for the selected node and all the nodes that are connected to the selected node. In this example, as shown in [Figure 7-3](#), if the Apply Node is selected, then a script will be generated for all the nodes that are connected to 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.

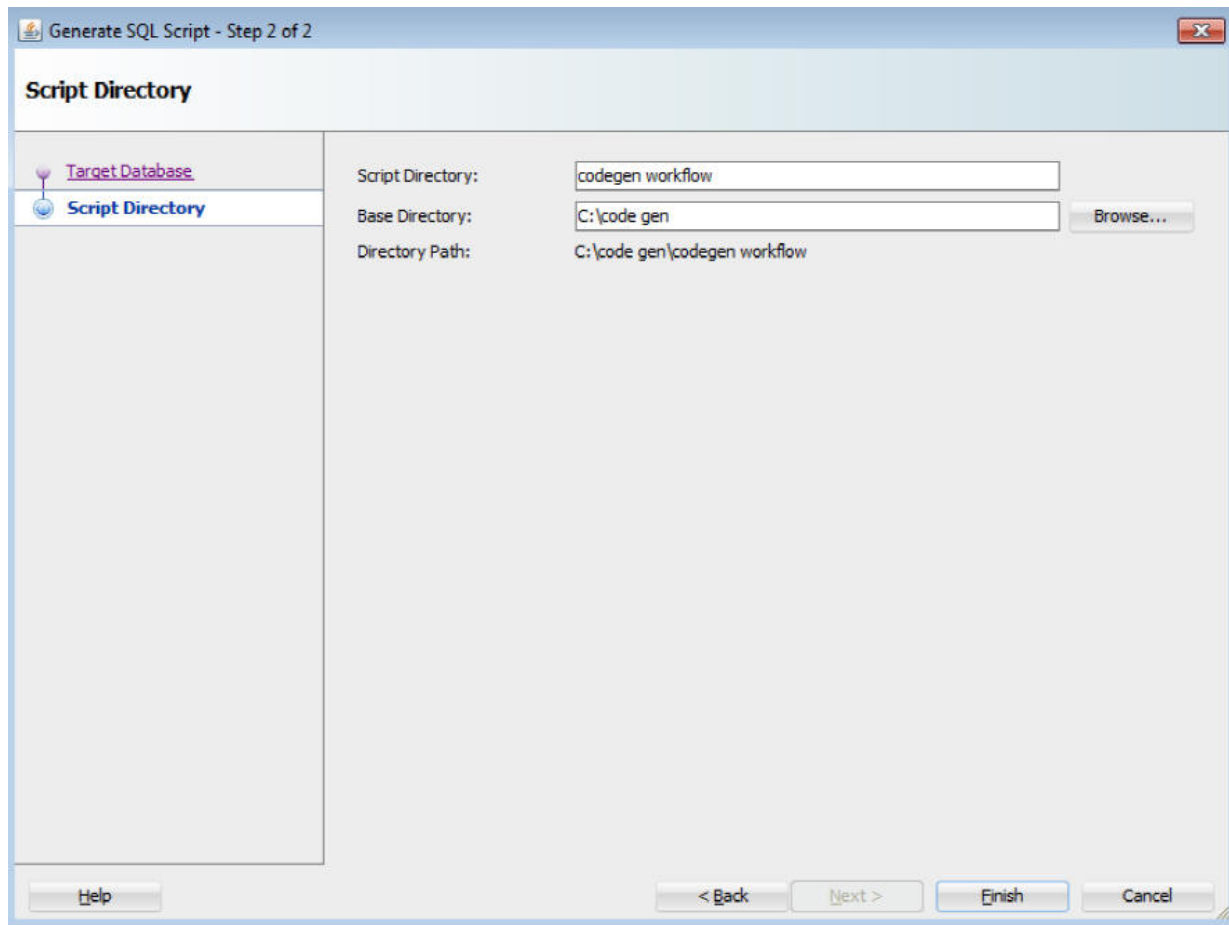
2. After selecting a Deploy options, the Generate SQL Script — Step 1 of 2 wizard opens, as shown in [Figure 7-4](#).

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



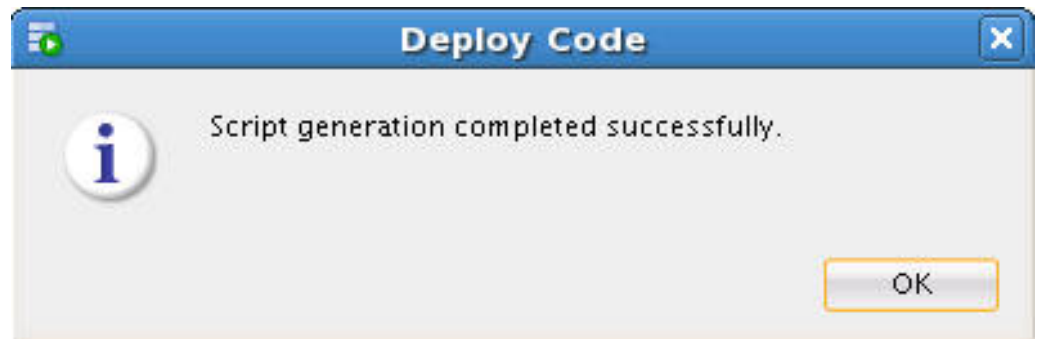
3. In the **Target Database Version** field, select the database version. Ensure that the generated script is compatible with the version of the database that you select here. This is the database where the SQL script will run.
4. Click **Next**. The Generate SQL Script — Step 2 of 2 window opens, as shown in [Figure 7-5](#).

Figure 7-5 Generate SQL Scripts - Step 2 of 2 Wizard



5. In the Step 2 of the Generate Script Wizard dialog box, 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.

- [List of Generated SQL Script Files](#)
The Structured Query Language (SQL) script files that are generated from the `codegen_workflow`, perform jobs that are related object cleanups, and other node specific operations.
- [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.
- [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.

List of Generated SQL Script Files

The Structured Query Language (SQL) script files that are generated from the `codegen_workflow`, perform jobs that are related object cleanups, and other node specific operations.

[Table 7-1](#) lists the SQL script files that are generated from the `codegen_workflow`, along with their descriptions:

Table 7-1 List of Generated Script Files and their Description

| Script File Type | Script File Name | Examples of Script Files Generated from the <code>codegen_workflow</code> | Description |
|------------------|-------------------------------------|--|--|
| Master Script | <code>workflow_name_Run.sql</code> | <code>codegen_workflow_Run.sql</code> | Invokes all the required node level scripts in the correct order. It performs the following tasks: <ul style="list-style-type: none"> Validates compatibility of the version of the script file with the Data Miner Repository version. Creates a workflow master table that contains entries for all the underlying objects created by the workflow script. |
| Cleanup Script | <code>workflow_name_Drop.sql</code> | <code>codegen_workflow_Drop.sql</code> | Drops all objects created by the master script. It drops the following: <ul style="list-style-type: none"> Hidden objects, such as tables generated for Explore Data nodes. Public objects such as model names created by the Build nodes. Tables created by the Create Table node. |
| Workflow Image | <code>workflow_name.png</code> | <code>codegen_workflow.png</code> | This is an image of the workflow at the time of script generation. |
| Node Script | <code>node_name.sql</code> | <ul style="list-style-type: none"> <code>Apply.sql</code> <code>Class_Build.sql</code> <code>Coefficients.sql</code> <code>INSUR_CUST_LTV_SAMPLE_APPLY.sql</code> <code>INSUR_CUST_LTV_SAMPLE_BUILD.sql</code> <code>MODEL_COEFFICIENTS.sql</code> <code>SCORED_CUSTOMER_S.sql</code> | 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. |

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.

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.
- [Structure of the Control Table](#)
The Control Table is created first by using the name specified in the control table name variable when the master script `workflow name_Run.sql` is run
- [Columns in the Control Table](#)
The columns in the Control Table contains information related to nodes and models.

Related Topics

- [Querying the Control Table](#)
After running the SQL scripts, you can query the Control Table to examine the generated objects.

Structure of the Control Table

The Control Table is created first by using the name specified in the control table name variable when the master script `workflow name_Run.sql` is run

The structure of the Control Table is as follows:

```
CREATE TABLE "&WORKFLOW_OUTPUT"  
(
```

```

NODE_ID VARCHAR2(30) NOT NULL,
NODE_NAME VARCHAR2(128) NOT NULL,
NODE_TYPE VARCHAR2(30) NOT NULL,
MODEL_ID VARCHAR2(30),
MODEL_NAME VARCHAR2(128),
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)
)

```

Columns in the Control Table

The columns in the Control Table contains information related to nodes and models.

[Table 7-2](#) lists the columns in the Control Table along with their description and examples.

Table 7-2 Columns in the Control Table and their Description

| Column Name | Description | Examples |
|-------------|---|---|
| NODE_ID | This is the ID of the node that constitutes a part of the workflow. It uniquely identifies the node. | 10001, 10002 |
| NODE_NAME | This is the name of the node that constitutes a part of the workflow. | Class Build, MINING_DATA_BUILD_V |
| NODE_TYPE | This is the category of node. | Data Source node, Class Build node and so on. |
| MODEL_ID | This is the ID of the workflow model. It uniquely identifies each model referenced within a workflow. | 10101, 10102 |
| MODEL_NAME | This is the name of the model. | CLAS_GLM_1_6 |
| MODEL_TYPE | Model type is the algorithm type used by the model. | Generalized Linear Model, Support Vector Machines and so on |

Table 7-2 (Cont.) Columns in the Control Table and their Description

| Column Name | Description | Examples |
|-----------------|--|--|
| OUTPUT_NAME | This is the name of the output. These are internally generated names unless the names are under the control of the user. | Table/View Name, Model Name, Text object names such as: ODMR\$15_37_21_839599RM AFRXI - table name "DMUSER"."CLAS_GLM_1_6" - fully qualified model name |
| OUTPUT_TYPE | It qualifies the type of output object. | Table, view, model |
| ADDITIONAL_INFO | This is the information that qualifies the purpose of the object about the script execution. | Target class for test lift result |
| CREATION_TIME | This is the time of object creation. | 11-DEC-12 03.37.25.935193000 PM (format determined by locale) |
| COMMENTS | Comment to qualify the role of the object about the script execution. | Output Data (displayed for nodes like Data Source) Data Usage (displayed for the view passed into model build) Weights Setting (displayed for a weights table passed into model build) Build Setting (displayed for a build settings table passed into model build) Model (displayed for a Model object) |

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](#)
Before scheduling workflow script files, certain tasks related to Oracle Database, SQL script files, Oracle Data Miner repository and Oracle Data Miner user account must be performed.
- [Schedule SQL Scripts Using SQL Developer](#)
Oracle SQL Developer provides the graphical user interface to define Scheduler Jobs.

- [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.

Prerequisites for Scheduling Workflow Script Files

Before scheduling workflow script files, certain tasks related to Oracle Database, SQL script files, Oracle Data Miner repository and Oracle Data Miner user account must be performed.

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.
- [Adding Complete Directory Path in the Master Script](#)
You can add the complete directory path in the master script `codegen_workflow_Run.sql` by editing the master script file.
- [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.

Adding Complete Directory Path in the Master Script

You can add the complete directory path in the master script `codegen_workflow_Run.sql` by editing the master script file.

To add the complete directory path in the master script:

1. Open the master script `codegen_workflow_Run.sql`, and locate the following lines:

```
-- 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:

```
-- 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.

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:
 - a. **Name**
 - b. 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.

Schedule SQL Scripts Using SQL Developer

Oracle SQL Developer provides the graphical user interface to define Scheduler Jobs.

Scheduling Structured Query Language (SQL) scripts using SQL Developer involves the following:

- [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.
- [Defining Scheduler Job using Job Wizard](#)
The Job wizard allows you to create a job schedule, using which you can define a workflow job

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:
 - a. **Name**
 - b. 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.

Defining Scheduler Job using Job Wizard

The Job wizard allows you to create a job schedule, using which you can define a workflow job

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.
3. Under **Scheduler**, right-click **Jobs** and click **New Jobs** (wizard). The Create Jobs dialog box opens.
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 from the drop-down list.
 - c. **Connect Credential Name**: Select the connection credential that you created 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.

Related Topics

- [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.

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.

Related Topics

- [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.

Deploying SQL Scripts on the Target Database

Deploying the SQL scripts involves running the master script file from the base directory.

To deploy SQL scripts, run the following master script file:

```
> @" C: base directory\workflow name_Run.sql "
```

```
> @" C:\<base directory>\workflow name_Run.sql"
```

For example, run the following master script `codegen_workflow_Run.sql` in SQLPlus from the base directory:

```
>@" 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 following master script:

```
>@" C:\code gen\codegen workflow\codegen_workflow_Drop.sql"
```

```
>@" C:\code gen\codegen workflow\codegen_workflow_Run.sql"
```

- **Querying the Control Table**
After running the SQL scripts, you can query the Control Table to examine the generated objects.

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 the following command in SQLPlus:

```
>select * from workflow_name
```

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

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

Figure 7-7 An Output Table - Result of the Query

| MODEL_ID | MODEL_NAME | MODEL_TYPE | MODEL_ID | MODEL_NAME | MODEL_TYPE | OUTPUT_NAME | OUTPUT_TYPE | ADDITIONAL_INFO | COMMENTS | |
|----------|-----------------------------|-------------------------|----------------------|----------------|------------------------|-------------------------------|-------------------------------|--------------------|----------------------|-------------|
| 1 | INSUR_CUST_LTY_SAMPLE BUILD | DataSourceNode | (null) | (null) | (null) | 00NB407_20_05_708363W807NLY | VIEW | (null) | Output Data | |
| 2 | INSUR_CUST_LTY_SAMPLE APPLY | DataSourceNode | (null) | (null) | (null) | 00NB407_20_06_802192M0V9379 | VIEW | (null) | Output Data | |
| 3 | Class Build | ClassificationBuildNode | (null) | (null) | (null) | 00NB407_20_08_034103E0MFA5W | VIEW | (null) | Input Data | |
| 4 | Class Build | ClassificationBuildNode | (null) | (null) | (null) | 00NB47523305700DMR4TMP | TABLE | (null) | Build Data | |
| 5 | Class Build | ClassificationBuildNode | (null) | (null) | (null) | 00NB49090666200DMR4TMP | TABLE | (null) | Test Data | |
| 6 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_08_170523RPH0W0W | VIEW | (null) | Data Usage | |
| 7 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_08_187014000000P | TABLE | (null) | Weights Setting | |
| 8 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_08_195131M02CGKI | TABLE | (null) | Build Setting | |
| 9 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | "DMUSER"."CLAS_SVM_MODEL" | MODEL | (null) | Model | |
| 10 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_09_614935E0PDKTL | TABLE | (null) | Apply Data | |
| 11 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_10_267728HJLXZ8 | TABLE | (null) | Test Metric | |
| 12 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_12_733203VLEVPKE | TABLE | (null) | Confusion Matrix | |
| 13 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_15_391757VTR1CTM | TABLE | EUY_INSURADUCE-No | lift Result | |
| 14 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_15_658637ITH0N0A | TABLE | EUY_INSURADUCE-Yes | lift Result | |
| 15 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_15_8402300DE0N0A | TABLE | EUY_INSURADUCE-No | R0C Result | |
| 16 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | .79952365159908403 | SCALAR | EUY_INSURADUCE-No | R0C Area Under Curve | |
| 17 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | 00NB407_20_16_589615B0PDKKI | TABLE | EUY_INSURADUCE-Yes | R0C Result | |
| 18 | Class Build | ClassificationBuildNode | 10025 | CLAS_SVM_MODEL | Support Vector Machine | .80123734474162129 | SCALAR | EUY_INSURADUCE-Yes | R0C Area Under Curve | |
| 19 | 10013 | Coefficients | ModelDetailsDataNode | (null) | (null) | (null) | 00NB407_20_18_474901W0R070A | VIEW | (null) | Output Data |
| 20 | MODEL_COEFFICIENTS | CreateTableNode | (null) | (null) | (null) | "DMUSER"."MODEL_COEFFICIENTS" | TABLE | (null) | Output Data | |
| 21 | 10018 | Apply | ApplyNode | (null) | (null) | (null) | 00NB407_20_20_536188M0V0M70 | VIEW | (null) | Output Data |
| 22 | 10021 | SCORED_CUSTOMERS | CreateTableNode | (null) | (null) | (null) | "DMUSER"."SCORED_CUSTOMERS_V" | VIEW | (null) | Output Data |

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.

Related Topics

- [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.
- [Structure of Control Table](#)
The Control Table is created first by using the name specified in the control table name variable when the master script `workflow name_Run.sql` is run

8

Using PL/SQL API to Manage Workflows

This chapter explains how to manage workflows using the PL/SQL APIs.

- [About PL/SQL APIs](#)
Oracle Data Miner ships with a set of repository PL/SQL APIs that enable applications to manage Oracle Data Miner projects and workflow directly.
- [PL/SQL APIs](#)
You can use the PL/SQL APIs to manage projects and workflows.
- [Repository Views](#)
In the Repository View, you can query information related to workflows and projects, and also monitor the status.
- [PL/SQL APIs Use Cases](#)
The PL/SQL API use cases demonstrate how to run the PL/SQL APIs to schedule and run a Build workflow and an Apply workflow.

About PL/SQL APIs

Oracle Data Miner 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

PL/SQL APIs

You can use the PL/SQL APIs to manage projects and workflows.

Use the PL/SQL APIs to perform the following tasks:

- [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.

- **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.
- **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.
- **WF_RUN**
The function `WF_RUN` that runs a workflow contains signatures that accepts names, project IDs, workflow and specific nodes to run.
- **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.
- **WF_RENAME**
The procedure `WF_RENAME` renames an existing workflow.
- **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.
- **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.
- **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.

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 8-1 List of Parameters for PROJECT_CREATE Function

| Parameter | Description |
|-----------------------------|---|
| <code>p_project_name</code> | Assign a name to the project that is created. |
| <code>p_comment</code> | Specify any comment that is to be applied to the project. |

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 8-2 List of Parameters for PROJECT_RENAME Procedure

| Parameters | Description |
|-----------------------------|--|
| <code>p_project_id</code> | Specify the project ID of the project to rename. |
| <code>p_project_name</code> | Specify the new name for the project. |

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 8-3 List of Parameters for PROJECT_DELETE procedure

| Parameters | Description |
|----------------------------|--|
| <code>p_project_id</code> | Specify the project ID of the project to delete. |
| <code>p_project_ids</code> | Specify the project IDs of the projects to delete. |

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.

- [WF_RUN Parameters](#)
You can execute the WF_RUN function using different parameter combinations:
- [WF_RUN with Project Name, Workflow Name, and Node Name](#)
The WF_RUN function with the project name, workflow name and node name parameters:
- [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:
- [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:
- [WF_RUN with Project ID, Workflow ID and Node IDs](#)
The WF_RUN function with the project ID, workflow ID and node ID parameters:

WF_RUN Parameters

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

| Parameters | Description |
|-----------------|--|
| P_PROJECT_NAME | Specify the project name that the workflow was created in. |
| P_PROJECT_ID | Specify the project ID that the workflow was created in. |
| P_WORKFLOW_NAME | Specify the workflow name to run. |
| P_WORKFLOW_ID | Specify the workflow ID to run. |
| P_NODE_NAMES | Specify the node names in the workflow to run. |
| P_NODE_IDS | Specify the node IDs in the workflow to run. |

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

| Parameters | Description |
|-------------------|---|
| P_RUN_MODE | <ul style="list-style-type: none"> VALIDATE_ONLY: Validates parent nodes of the specified nodes. 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. RERUN_NODE_ONLY: Resets the status of the specified nodes to READY state. Then these nodes are run again. RERUN_NODE_CHILDREN: Resets the status of the specified nodes and their children nodes to READY state. Then these nodes are run again. RERUN_NODE_PARENTS: Resets the status of the specified nodes and their parent nodes to READY state. Then these nodes are run again. RERUN_WORKFLOW: Resets the status of all nodes to READY state. Then the nodes are run (complete workflow run). Note: p_node_names is ignored. |
| 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. |

WF_RUN with Project Name, Workflow Name, and Node Name

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

```
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
```

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:

```
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_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
```

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
```

WF_RUN with Project ID, Workflow ID and Node IDs

The WF_RUN function with the project ID, workflow ID and node ID 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_SCHEDULE IN VARCHAR2 DEFAULT NULL,
                                P_JOB_CLASS IN VARCHAR2 DEFAULT NULL)
RETURN VARCHAR
```

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 8-5 List of Parameters for WF_STOP Procedure

| Parameter | Description |
|----------------------------|--|
| <code>p_workflow_id</code> | Specify the workflow ID of the workflow to cancel. |

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 8-6 List of Parameters for WF_RENAME Procedure

| Parameters | Description |
|------------------------------|--|
| <code>p_workflow_id</code> | Specify the workflow ID to rename. |
| <code>p_workflow_name</code> | Specify the new workflow name. |
| <code>p_mode</code> | This parameter is for internal use only. |

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:

```
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

| Parameter | Description |
|----------------------------|---|
| <code>p_workflow_id</code> | Specify the ID of the workflow that is to be deleted. |

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:

```
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

| Parameters | Description |
|------------------------------|--|
| <code>p_project_id</code> | Specify the ID of the project in to which the workflow will be imported. |
| <code>p_workflow_name</code> | Specify the workflow to import. |

Table 8-8 (Cont.) List of Parameters for WF_IMPORT Function

| Parameters | Description |
|-----------------|---|
| p_workflow_data | Specify the workflow metadata. This workflow should be previously exported by the Oracle Data Miner and the workflow version should not be newer than what the Repository supports. |
| p_comment | Specify the comment to be applied to the workflow. |
| p_force | Determines whether to force import if the workflow has object name conflicts with existing workflows in the repository. The applicable values are: <ul style="list-style-type: none"> 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. |

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 8-9 List of Parameters for WF_EXPORT Function

| Parameters | Description |
|---------------|---|
| p_workflow_id | Specify the ID of the workflow to export. |

Related Topics

- [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.

Repository Views

In the Repository View, you can query information related to workflows and projects, and also monitor the status.

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**
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.
- **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.
- **ODMR_USER_WORKFLOW_LOG**
You can query the logs of workflow run by using the `ODMR_USER_WORKFLOW_LOG` repository view.
- **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.
- **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.
- **ODMR_USER_WF_CLAS_TEST_RESULTS**
By using the `ODMR_USER_WF_CLAS_TEST_RESULTS` repository view, you can query the generated classification results for a specific mining model in the last workflow run.
- **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.
- **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.
- **ODMR_USER_WORKFLOW_ALL_POLL**
You can use the `ODMR_USER_WORKFLOW_ALL_POLL` view the status of Oracle Data Miner workflows, and thereby minimize the use of Oracle Scheduler views.

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 8-10 ODMR_USER_PROJECT_WORKFLOW Repository View

| Column | Data Type | Description |
|------------|-----------|---|
| PROJECT_ID | NUMBER | This is the ID of the project in which the workflow was created in. |

Table 8-10 (Cont.) ODMR_USER_PROJECT_WORKFLOW Repository View

| Column | Data Type | Description |
|----------------------|----------------------|---|
| PROJECT_NAME | VARCHAR2 (128 CHAR) | This is the name of the project in which the workflow was created in. |
| PJ_CREATION_TIME | TIMESTAMP (6) | This is the project creation time. |
| PJ_LAST_UPDATED_TIME | TIMESTAMP (6) | Project last modified time stamp. |
| PJ_COMMENTS | VARCHAR2 (4000 CHAR) | These are comments related to the project, if any. |
| WORKFLOW_ID | NUMBER | This is the workflow ID. |
| WORKFLOW_NAME | VARCHAR2 (128 CHAR) | This is the name of the workflow. |
| WORKFLOW_DATA | XMLTYPE | This is the workflow metadata in XML format. |
| CHAIN_NAME | VARCHAR (128 CHAR) | This is for internal use only. |
| STATUS | VARCHAR (30 CHAR) | <ul style="list-style-type: none"> • INACTIVE: Indicates that the workflow is idle. • ACTIVE: Indicates that the workflow is running. • QUEUED: Indicates that the workflow is in queue to run. • STOPPING: Indicates that the workflow is being stopped. • STOPPED: Indicates that the running workflow has stopped. • SCHEDULED: Indicates that the workflow is scheduled to run. |
| WF_CREATION_TIME | TIMESTAMP (6) | This is the workflow creation time stamp. |
| WF_LAST_UPDATED_TIME | TIMESTAMP (6) | This is the workflow last modified time stamp. |
| WF_COMMENTS | VARCHAR2 (4000 CHAR) | These are comments related to the workflow, if any. |

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

| Column | Data Type | Description |
|-----------------|--------------------------------|---|
| WORKFLOW_ID | NUMBER | This is the ID of the workflow. |
| WF_JOB_NAME | VARCHAR2 (261) | This is the Scheduler Job that runs the workflow. |
| LOG_DATE | TIMESTAMP (6) WITH TIME ZONE | This is the log entry time stamp. |
| LOG_ID | NUMBER | This is the log entry ID. |
| NODE_ID | VARCHAR2 (261) | This is the workflow node ID. |
| SUBNODE_ID | VARCHAR2 (261) | This is the workflow sub node. For example, a Model node in a Build node. |
| NODE_STATUS | VARCHAR2 (11) | <ul style="list-style-type: none"> RUNNING: Indicates that the node is running. SUCCEEDED: Indicates that the node run has completed successfully. FAILED: Indicates that the node has failed to complete. NOT_STARTED: Indicates that the node is waiting to run. SCHEDULED: Indicates that the node is scheduled to run. PAUSED: Indicates that the node is paused. This is an exception. STOPPED: Indicates that the node run has stopped. STALLED: Indicates that the node has stalled. This is an exception. |
| SUBNODE_STATUS | VARCHAR2 (30) | Same as Node status |
| NODE_START_TIME | TIMESTAMP (6) WITH TIME ZONE | This is the workflow node start time stamp. |
| NODE_RUN_TIME | INTERVAL DAY (9) TO SECOND (6) | This is the node run time. |
| ERROR_CODE | NUMBER | This is the error code returned by the node. |
| LOG_MESSAGE | VARCHAR2 (4000 CHAR) | This is the log message generated by the node. |

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 8-12 ODMR_USER_WORKFLOW_LOG Repository Views

| Column | Data Type | Description |
|---------------------|--------------------------------|---|
| LOG_ID | NUMBER | This is the log entry ID. |
| JOB_NAME | VARCHAR2 (128 CHAR) | This is the Scheduler Job that runs the workflow. |
| PROJ_NAME | VARCHAR2 (128 CHAR) | This is the project in which the workflow was created in. |
| PRO_ID | NUMBER | This is the project ID in which the workflow was created in. |
| WF_NAME | VARCHAR2 (128 CHAR) | This is the workflow name. |
| WF_ID | NUMBER | This is the workflow ID. |
| NODE_NAME | VARCHAR2 (128 CHAR) | This is the name of the node in the workflow. |
| NODE_ID | VARCHAR2 (30) | This is the node ID. |
| SUBNODE_NAME | VARCHAR2 (128 CHAR) | This is the workflow sub node name. For example, Model name in a Build node. |
| SUBNODE_ID | VARCHAR2 (30) | This is the workflow sub node ID. for example, the Model ID in a Build Node. |
| LOG_TIMESTAMP | TIMESTAMP (6) WITH TIME ZONE | This is the log entry time stamp. |
| LOG_DURATION | INTERVAL DAY (3) TO SECOND (0) | This is the log entry duration in days and seconds. |
| LOG TYPE | VARCHAR2 (30 CHAR) | <ul style="list-style-type: none"> • WARN: Indicates warning. • ERR: Indicates error. • INFO: Indicates informational content. |
| LOG SUBTYPE | VARCHAR2 (30 CHAR) | <ul style="list-style-type: none"> • START: Indicates start of a task. • END: Indicates end of a task. |
| LOG_MESSAGE | NVARCHAR2 (2000) | This is the log message generated by the node. |
| LOG_MESSAGE_DETAILS | VARCHAR2 (4000 CHAR) | This is the log message details generated by the node. |

Table 8-12 (Cont.) ODMR_USER_WORKFLOW_LOG Repository Views

| Column | Data Type | Description |
|----------|--------------------|---|
| LOG_TASK | VARCHAR2 (30 CHAR) | <p>When a node is running, it performs one or more of the following tasks:</p> <ul style="list-style-type: none"> • PROJECT • WORKFLOW • NODE • SUBNODE • VALIDATE • SAMPLE • CACHE • STATISTICS • FEATURES • DATAPREP • BUILD • TEST • APPLY • TRANSFORM • TEXT • BUILDTEXT • APPLYTEXT • OUTPUT • CLEANUP • CREATE EXPLORE STATISTICS • CREATE HISTOGRAM • CREATE SAMPLE DATA • CREATE HISTOGRAM SAMPLE • CREATE DATA GUIDE • CREATE PROJECT • DELETE PROJECT • RENAME PROJECT • SET COMMENT • CREATE WORKFLOW • RUN WORKFLOW • RENAME WORKFLOW • DELETE WORKFLOW • IMPORT WORKFLOW • EXPORT WORKFLOW |

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

| Column | Data Types | Description |
|---------------|---------------------|--|
| PROJECT_ID | NUMBER | This is the ID of the project in which the workflow was created in. |
| PROJECT_NAME | VARCHAR2 (128 CHAR) | This is the name of the project in which the workflow was created in. |
| WORKFLOW_ID | NUMBER | This is the ID of the workflow. |
| WORKFLOW_NAME | VARCHAR2 (128 CHAR) | This is the name of the workflow. |
| NODE_TYPE | VARCHAR2 (30 CHAR) | This is the node type. |
| NODE_NAME | VARCHAR2 (128 CHAR) | This is the name of the node. |
| NODE_ID | NUMBER | This is the ID of the node. |
| NODE_STATUS | VARCHAR2 (10 CHAR) | <ul style="list-style-type: none"> INVALID: Indicates that the node is not valid, and it cannot be run. WARNING: Indicates that the node has run but with warning. READY: Indicates that the node is ready to run. FAILURE: Indicates that the node run has failed. COMPLETE: Indicates that the node run has completed successfully. |

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

| Column | Data Types | Description |
|---------------|---------------------|---|
| PROJECT_ID | NUMBER | This is the project ID in which the workflow was created in. |
| PROJECT_NAME | VARCHAR2 (128 CHAR) | This is the name of the project in which the workflow was created in. |
| WORKFLOW_ID | NUMBER | This is the ID of the workflow. |
| WORKFLOW_NAME | VARCHAR2 (128 CHAR) | This is the name of the workflow. |
| NODE_TYPE | VARCHAR2 (30 CHAR) | This is the node type. |
| NODE_ID | NUMBER | This is the workflow node ID. |

Table 8-14 (Cont.) ODMR_USER_WORKFLOW_MODELS Repository Views

| Column | Data Types | Description |
|--------------------|---------------------|---|
| NODE_NAME | VARCHAR2 (128 CHAR) | This is the name of the workflow node. |
| NODE_STATUS | VARCHAR2 (30 CHAR) | <ul style="list-style-type: none"> INVALID: Indicates that the node is not valid, and cannot be run. WARNING: Indicates that the node has run but with warning. READY: Indicates that the node is ready to run. FAILURE: Indicates that the node run has failed. COMPLETE: Indicates that the node run has completed successfully. |
| MODEL_TYPE | VARCHAR2 (30 CHAR) | This is the model type. For example, Naive Bayes Model. |
| MODEL_ID | NUMBER | This is the ID of the model. |
| MODEL_NAME | VARCHAR2 (128 CHAR) | This is the name of the model. |
| MODEL_STATUS | VARCHAR2 (30 CHAR) | Same as node status. |
| MODEL_CREATIONDATE | VARCHAR2 (30 CHAR) | This is the date when the model is created. |

ODMR_USER_WF_CLAS_TEST_RESULTS

By using the `ODMR_USER_WF_CLAS_TEST_RESULTS` repository view, you can query the generated classification results for a specific mining model in the last workflow run.

[Table 8-15](#) provides more information about this .

Table 8-15 ODMR_USER_WF_CLAS_TEST_RESULTS

| Column | Data Type | Description |
|---------------|---------------------|---|
| PROJECT_ID | NUMBER | This is the ID of the project in which the workflow was created in. |
| PROJECT_NAME | VARCHAR2 (128 CHAR) | This is the name of the project in which the workflow was created in. |
| WORKFLOW_ID | NUMBER | This is the ID of the workflow. |
| WORKFLOW_NAME | VARCHAR2 (128 CHAR) | This is the name of the workflow. |
| NODE_TYPE | VARCHAR2 (30 CHAR) | This is the node type. |
| NODE_ID | NUMBER | This is the ID of the node. |

Table 8-15 (Cont.) ODMR_USER_WF_CLAS_TEST_RESULTS

| Column | Data Type | Description |
|---------------------|------------------------|---|
| NODE_NAME | VARCHAR2 (128 CHAR) | This is the name of the node. |
| NODE_STATUS | VARCHAR2 (10 CHAR) | <ul style="list-style-type: none"> 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 (128 CHAR) | This is the name of the model. |
| MODEL_STATUS | VARCHAR2 (10 CHAR) | <ul style="list-style-type: none"> 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. |
| RESULT_CREATIONDATE | VARCHAR2 (30 CHAR) | This is the creation date of the result. |
| 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 | <p>The table of DM_NESTED_CATEGORICAL, where:</p> <ul style="list-style-type: none"> ATTRIBUTE_NAME contains target class VALUE contains lift result table |

Table 8-15 (Cont.) ODMR_USER_WF_CLAS_TEST_RESULTS

| Column | Data Type | Description |
|----------|------------------------|--|
| ROCS | DM_NESTED_CATEGORICALS | The table of DM_NESTED_CATEGORICAL, where: <ul style="list-style-type: none"> ATTRIBUTE_NAME contains target class VALUE contains ROC result table |
| ROC_AREA | DM_NESTED_NUMERICALS | The table of DM_NESTED_NUMERICAL, where: <ul style="list-style-type: none"> ATTRIBUTE_NAME contains target class VALUE contains ROC area under curve value |

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 8-16 ODMR_USER_WF_REGR_TEST_RESULTS Repository Views

| Column | Data Type | Description |
|---------------|---------------------|---|
| PROJECT_ID | NUMBER | This is the ID of the project under which the workflow is created. |
| PROJECT_NAME | VARCHAR2 (128 CHAR) | This is the name of the project under which the workflow is created. |
| WORKFLOW_ID | NUMBER | This is the workflow ID. |
| WORKFLOW_NAME | VARCHAR2 (128 CHAR) | This is the name of the workflow. |
| NODE_TYPE | VARCHAR2 (30 CHAR) | This is the type of the node. For example, Build node, Model node, and so on. |
| NODE_ID | NUMBER | This is the ID of the node. |
| NODE_NAME | VARCHAR2 (128 CHAR) | This is the name of the node. |

Table 8-16 (Cont.) ODMR_USER_WF_REGR_TEST_RESULTS Repository Views

| Column | Data Type | Description |
|---------------------|---------------------|---|
| NODE_STATUS | VARCHAR2 (10 CHAR) | <ul style="list-style-type: none"> 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 (128 CHAR) | This is the name of the model. |
| MODEL_STATUS | VARCHAR2 (10 CHAR) | <ul style="list-style-type: none"> 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. |
| RESULT_CREATIONDATE | VARCHAR2 (30 CHAR) | This is the creation date of the result. |
| TEST_METRICS | VARCHAR2 (128 CHAR) | This is the test metrics result table that contains Predictive Confidence, Root Mean Square error, and so on. |
| RESIDUAL_PLOT | VARCHAR2 (128 CHAR) | This is the test Residual Plot table. |

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

| Column | Data Type | Description |
|---------------|---------------------|---|
| PROJECT_ID | NUMBER | The project ID of the project under which the workflow is created. |
| PROJECT_NAME | VARCHAR2 (128 CHAR) | The name of the project under which the workflow is created. |
| WORKFLOW_ID | NUMBER | This is the workflow ID. |
| WORKFLOW_NAME | VARCHAR2 (128 CHAR) | This is the name of the workflow. |
| NODE_TYPE | VARCHAR2 (30 CHAR) | This is the type of the node. For example, Build node, Model node and so on. |
| NODE_ID | NUMBER | This is the ID of the node. |
| NODE_NAME | VARCHAR2 (128 CHAR) | This is the name of the node. |
| NODE_STATUS | VARCHAR2 (10 CHAR) | <ul style="list-style-type: none"> 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 (128 CHAR) | This is the name of the model. |
| MODEL_STATUS | VARCHAR2 (10 CHAR) | <ul style="list-style-type: none"> 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. |

Table 8-17 (Cont.) ODMR_USER_WF_TEST_RESULTS Repository Views

| Column | Data Type | Description |
|---------------------|------------------------|--|
| RESULT_CREATIONDATE | VARCHAR2 (30 CHAR) | This is the creation date of the result. |
| TEST_METRICS | VARCHAR2 (128 CHAR) | This is the test metrics result table that contains Predictive Confidence, Root Mean Square error and so on. |
| CONFUSION_MATRIX | VARCHAR2 (128 CHAR) | This is the test Confusion Matrix result table. |
| LIFTS | DM_NESTED_CATEGORICALS | Table of DM_NESTED_CATEGORICAL, where: <ul style="list-style-type: none"> ATTRIBUTE_NAME contains target class VALUE contains lift result table |
| ROCS | DM_NESTED_CATEGORICALS | Table of DM_NESTED_CATEGORICAL, where: <ul style="list-style-type: none"> ATTRIBUTE_NAME contains target class VALUE contains ROC result table |
| ROC_AREA | DM_NESTED_NUMERICALS | Table of DM_NESTED_NUMERICAL, where: <ul style="list-style-type: none"> ATTRIBUTE_NAME contains target class VALUE contains ROC area under curve value |
| RESIDUAL_PLOT | VARCHAR2 (128 CHAR) | This is the test Residual Plot table. |

ODMR_USER_WORKFLOW_ALL_POLL

You can use the ODMR_USER_WORKFLOW_ALL_POLL view the status of Oracle Data Miner workflows, and thereby minimize the use of Oracle Scheduler views.

However, you must use Oracle Scheduler view to determine the status of workflow jobs. The ODMR_USER_WORKFLOW_ALL_POLL repository view can provide complete information about the status of failed workflows. The repository view ODMR_USER_WORKFLOW_ALL_POLL for workflow status polling contains the following repository properties:

- POLLING_IDLE_RATE**: Determines the rate at which the client will poll the database when there are no workflows detected as running.
- POLLING_ACTIVE_RATE**: Determines the rate at which the client will poll the database when there are workflows detected running.

- `POLLING_IDLE_ENABLED`: Determines automatic updates to scheduled workflow jobs.
- `POLLING_COMPLETED_WINDOW`: Determines the time required to include completed workflows in the polling query result.
- `PURGE_WORKFLOW_SCHEDULER_JOBS`: Purges old Oracle Scheduler objects generated by the running of Data Miner workflows.
- `PURGE_WORKFLOW_EVENT_LOG`: Controls how many workflow runs are preserved for each workflow in the event log. The events of the older workflow are purged to keep within the limit.

Related Topics

- [Summary of Oracle Data Miner Repository Properties for System Management](#)
A summary of the system management properties is available in the Oracle Data Miner repository.

PL/SQL APIs Use Cases

The PL/SQL API use cases demonstrate how to run the PL/SQL APIs to schedule and run a Build workflow and an Apply workflow.

The use cases are:

- [Premise of the PL/SQL Use Cases](#)
The PL/SQL API use case is built on two predefined workflows, which are available in the SQL Developer installation location.
- [Use case to Schedule and Run Apply Workflows](#)
This use case demonstrates how to run a lineage of the APPLY workflow, which is scheduled to run the `WF_RUN_API`.
- [Use Case to Schedule and Run a Build Workflow](#)
This use case demonstrates how to run the `WF_RUN_API` in a lineage of a workflow, poll the status and print node failure from the event log.

Premise of the PL/SQL Use Cases

The PL/SQL API use case is built on two predefined workflows, which are available in the SQL Developer installation location.

Predefined Workflows

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.

Location of the Demonstration 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_home\dataminer\demos\workflows`.

Use case to Schedule and Run Apply Workflows

This use case demonstrates how to run a lineage of the APPLY workflow, which is scheduled to run the WF_RUN_API.

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 the example, 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 8-1 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 => NULL,
                                p_end_date => NULL));
    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', 'RUNNING')) THEN
            DBMS_LOCK.SLEEP(10); -- wait for 10 secs
        ELSE
            EXIT; -- workflow run completes
        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;

```

- [Querying Scoring Result](#)
You can query the scoring result after the workflow run completes successfully.

Related Topics

- [WF_RUN](#)
You can execute the WF_RUN function using different parameter combinations:
- [Premise of the PL/SQL Use Cases](#)
The PL/SQL API use case is built on two predefined workflows, which are available in the SQL Developer installation location.

Querying Scoring Result

You can query the scoring result after the workflow run completes successfully.

To query the scoring results:

```
SELECT * FROM SCORED_CUSTOMERS_TBL
```

The output of the query is displayed in a table, as shown in [Figure 8-1](#).

Figure 8-1 Query Output for Scoring Results

| CLAS_SVM_MODEL_2_PRED | CLAS_SVM_MODEL_2_PROB | CUSTOMER_ID |
|-----------------------|-----------------------|-------------|
| No | 0.7989923820193029 | CU7854 |
| No | 0.8644111916016625 | CU12993 |
| No | 0.5759455906776023 | CU608 |
| No | 0.6650873302480882 | CU10025 |
| No | 0.9945940382328219 | CU11680 |
| No | 0.9434015317919479 | CU10706 |
| No | 0.7732896982616732 | CU6752 |
| No | 0.878879260306943 | CU6932 |
| No | 0.7094964165297339 | CU9555 |
| Yes | 0.7240977065353436 | CU475 |

Use Case to Schedule and Run a Build Workflow

This use case demonstrates how to run the WF_RUN_API in a lineage of a workflow, poll the status and print node failure from the event log.

You can run the `WF_RUN_API` in a lineage of a workflow, using the run modes `RERUN_NODE_ONLY` or `RERUN_NODE_PARENTS`.

The methods to run the `WF_RUN_API` using the two run modes 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 8-2](#), does the following:

- 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 8-2 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(128) := 'Project';
    v_workflow_name VARCHAR2(128) := '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 => NULL,
        p_end_date => NULL);
    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', 'RUNNING')) 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;
    DBMS_OUTPUT.PUT_LINE('Message: '||wf_log.log_message);
    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;

```

- [Querying the MODEL_COEFFICIENT Table](#)
You can fetch data from the MODEL_COEFFICIENTS table by running a query.
- [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.
- [Querying Test Results](#)
You can query test results for Confusion Matrix and Test Metrics from ODMR_USER_WF_CLAS_TEST_RESULTS and ODMR_USER_WF_REGR_TEST_RESULTS repository views.

Related Topics

- [WF_RUN](#)
You can execute the WF_RUN function using different parameter combinations:
- [WF_RUN Function](#)
The WF_RUN function with the name parameters and start date and end date:
- [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.

Querying the MODEL_COEFFICIENT Table

You can fetch data from the MODEL_COEFFICIENTS table by running a query.

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

| MODEL_SCHEMA | MODEL_NAME | CLASS | ATTRIBUTE_NAME | ATTRIBUTE_SUBNAME | ATTRIBUTE_VALUE | COEFFICIENT |
|--------------|------------------|-------|-------------------------|-------------------|-----------------|-----------------------|
| DMUSER | CLAS_SVM_MODEL_2 | Yes | AGE | (null) | (null) | -0.53817011175361995 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | (null) | (null) | (null) | -2.4125349971707002 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | T_AMOUNT_AUTOM_PAYMENTS | (null) | (null) | -0.45125800400101101 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | TIME_AS_CUSTOMER | (null) | 5 | 0.085648486023984005 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | TIME_AS_CUSTOMER | (null) | 4 | 0.114722297194206 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | TIME_AS_CUSTOMER | (null) | 3 | -0.060500984045086498 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | TIME_AS_CUSTOMER | (null) | 2 | -0.24627021266815999 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | TIME_AS_CUSTOMER | (null) | 1 | 0.106400413495056 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | STATE | (null) | WI | -0.074409405659113304 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | STATE | (null) | WA | -0.070309126558766394 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | STATE | (null) | UT | -0.51127437327744996 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | STATE | (null) | TX | 0.36160360293990002 |
| DMUSER | CLAS_SVM_MODEL_2 | Yes | STATE | (null) | OR | -0.245338212232384 |

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 8-3](#).

Figure 8-3 Query Output for Named Objects

| MODEL_NAME | MINING_FUNCTION | ALGORITHM | CREATION_DATE | BUILD_DURATION | MODEL_SIZE | COMMENTS |
|------------------|-----------------|-------------------------|---------------|---------------------|------------|----------|
| CLAS_SVM_MODEL_2 | CLASSIFICATION | SUPPORT_VECTOR_MACHINES | 30-NOV-14 | 0.99999999999999... | 0.0773 | BALANCED |

Querying Test Results

You can query test results for Confusion Matrix and Test Metrics from ODMR_USER_WF_CLAS_TEST_RESULTS and ODMR_USER_WF_REGR_TEST_RESULTS repository views.

You can query the test results from:

- ODMR_USER_WF_CLAS_TEST_RESULTS Repository View
- ODMR_USER_WF_REGR_TEST_RESULTS Repository View

This section contains examples to query the following:

- Query Test Metrics and Confusion Matrix results: [Example 8-3](#)
- Query Test Metrics: [Example 8-4](#)
- Query Confusion Matrix: [Example 8-5](#)
- Query Lift Table results: [Example 8-6](#)

Example 8-3 Querying Test Metrics and Confusion Matrix Results

```
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.

Figure 8-4 Query Output for Test Metrics and Confusion Matrix

| TEST_METRICS | CONFUSION_MATRIX |
|------------------------------|------------------------------|
| ODMR\$18_51_18_106346IFHRNMF | ODMR\$18_51_17_954530VMUXPWL |

Example 8-4 Querying TEST_METRICS

```
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.

Figure 8-5 Query Output for TEST METRICS

| METRIC_NAME | METRIC_VARCHAR_VALUE | METRIC_NUM_VALUE |
|-----------------------|------------------------------|---|
| MODEL_SCHEMA | DMUSER | (null) |
| MODEL_NAME | CLAS_SVM_MODEL_2 | (null) |
| TEST_DATA_NAME | ODMR\$18_51_13_674501NHLUMHU | (null) |
| MINING_FUNCTION | CLASSIFICATION | (null) |
| TARGET_ATTRIBUTE | BUY_INSURANCE | (null) |
| LEAST_TARGET_VALUE | Yes | (null) |
| ACCURACY | (null) | 71.7821782178217821782178217821782178217821782178 |
| TEST_ROWS | (null) | 404 |
| AVG_ACCURACY | (null) | 73.8824870203874889198429783462074205395 |
| PREDICTIVE_CONFIDENCE | (null) | 47.764974040774977839685956692414841079 |

Example 8-5 Querying CONFUSION_MATRIX

```
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.

Figure 8-6 Query Output for CONFUSION_MATRIX

| ACTUAL_TARGET_VALUE | PREDICTED_TARGET_VALUE | VALUE |
|---------------------|------------------------|-------|
| No | No | 207 |
| Yes | No | 23 |
| Yes | Yes | 83 |
| No | Yes | 91 |

Example 8-6 Querying Lift Result Table from CLAS_SVM_MODEL_2

```
SELECT
MODEL_NAME, a.ATTRIBUTE_NAME "target value", a.VALUE "lift result table"
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.

Figure 8-7 Query Output for Lift Result Table from CLAS_SVM_MODEL_2

| MODEL_NAME | target value | lift result table |
|------------------|--------------|------------------------------|
| CLAS_SVM_MODEL_2 | Yes | ODMR\$18_51_18_303812SPBWIYG |

Related Topics

- [Repository Views](#)

In the Repository View, you can query information related to workflows and projects, and also monitor the status.

A

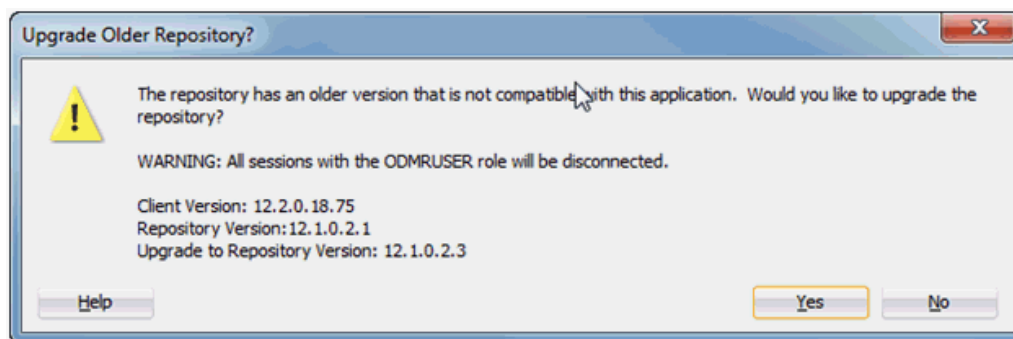
Oracle Data Miner Releases

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.

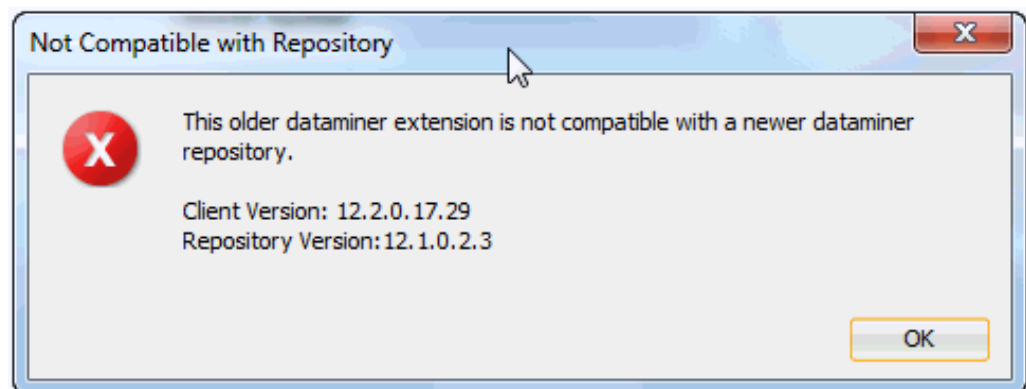
For example, if a newer client attempts to connect to an older repository, then the error shown in [Figure A-1](#) is displayed.

Figure A-1 Older Oracle Data Miner Repository Error



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



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** — When you select the SQL Developer Help menu, then in the **About** tab you can view the build and version number of Oracle SQL Developer.
- **Oracle Data Miner Extension** — When you select the SQL Developer Help menu, then in the **Extensions** tab you can view the Oracle Data Miner version number.
- **Oracle Data Miner Repository Version** — You can find the repository version by running the query shown in [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 *Oracle 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 A-1 SQL Developer Oracle Data Miner Extension and Oracle Data Miner Repository Compatibility

| SQL Developer | Data Miner Extension | Data Miner Repository Version | Data Miner XML Schema Version | Release Type (Full or Path) | Limitations |
|--|----------------------|-------------------------------|-------------------------------|---|-------------------------------------|
| SQL Developer 3.0 EA4 3.0.03 Build MAIN-03.97 | 11.2.0.1.9.96 | 11.2.0.1.9 | 11.2.0.1.9 | EA4 (only EA release to include Data Miner) | Does not work on 11.2.0.3 and later |
| SQL Developer 3.0 RTM 3.0.04 Build MAIN-04.34 | 11.2.0.1.10.109 | 11.2.0.1.10 | 11.2.0.1.9 | RTM (released 03/30/11) | Does not work on 11.2.0.3 or later |
| SQL Developer 3.0 | 11.2.0.2.04.38 | 11.2.0.1.12 | 11.2.0.1.9 | Patch (Dev internal only) | Does not work on 11.2.0.3 or later |
| SQL Developer 3.0 | 11.2.0.2.04.39 | 11.2.0.1.13 | 11.2.0.1.9 | Patch (internal) | Does not work on 11.2.0.3 or later |
| SQL Developer 3.0 | 11.2.0.2.04.40 | 11.2.0.1.13 | 11.2.0.1.9 | Patch (external) link to patch | Does not work on 11.2.0.3 or later |
| SQL Developer 3.1 EA1 | 11.2.1.1.xx.xx | 11.2.1.1.1 | 11.2.1.1.1 | EA1 candidate (not released) | None |
| SQL Developer 3.1 EA1 3.1.05 Build MAIN-05.97 | 11.2.1.1.05.97 | 11.2.1.1.2 | 11.2.1.1.1 | EA1 (released 10/10/11) | None |

Table A-1 (Cont.) SQL Developer Oracle Data Miner Extension and Oracle Data Miner Repository Compatibility

| SQL Developer | Data Miner Extension | Data Miner Repository Version | Data Miner XML Schema Version | Release Type (Full or Path) | Limitations |
|---|----------------------|-------------------------------|-------------------------------|--|--|
| SQL Developer 3.1 | 11.2.1.1.06.44 | 11.2.1.1.3 | 11.2.1.1.1 | EA2 (released 11/15/11) | None |
| SQL Developer 3.1 | 11.2.1.06.82 | 11.2.1.1.4 | 11.2.1.1.1 | EA3 (released 12/21/11) | None |
| SQL Developer 3.1 RTM 3.1.07 Build MAIN-07.42 | 11.2.1.1.07.42 | 11.2.1.1.5 | 11.2.1.1.1 | RTM (released 02/07/12) | None |
| SQL Developer 3.2 RTM Version 3.2.09 Build MAIN-09-18 | 11.2.1.1.09.18 | 11.2.1.1.6 | 11.2.1.1.1 | RTM (released 08/20/12) | None |
| SQL Developer 3.2.1 RTM Version 3.2.10.09 Build MAIN-09-57 | 11.2.1.1.09.57 | 11.2.1.1.6 | 11.2.1.1.1 | RTM (released 09/24/12). For Data Miner, there were no changes made between 3.2 and 3.2.1. | None |
| SQL Developer 3.2.2 RTM Version 3.2.20.09 Build MAIN-09.87 | 11.2.1.1.09.87 | 11.2.1.6 | 11.2.1.1.1 | RTM (released 11/01/12). For Data Miner, there were no changes made between 3.2 and 3.2.2. | None |
| SQL Developer 4.0 EA1 (was 3.3) Version 4.0.0.12 Build MAIN-12.27 | 11.2.2.1.12.27 | 11.2.2.1.1 | 11.2.2.1.1 | EA1 (released 07/11/13) | None |
| SQL Developer 4.0 EA2 Version 4.0.0.12 Build MAIN-12.84 | 12.2.0.12.84 | 12.1.0.1.1 | 12.1.0.1.1 | EA2 (released 09/13/13) | None |
| SQL Developer 4.0 EA3 Version 4.0.0.13 Build MAIN-13.30 | 12.2.0.13.30 | 12.1.0.1.2 | 12.1.0.1.2 | EA3 (released 11/05/13) | Migration from prior versions is broken. Workaround documented in OTN posting. |

Table A-1 (Cont.) SQL Developer Oracle Data Miner Extension and Oracle Data Miner Repository Compatibility

| SQL Developer | Data Miner Extension | Data Miner Repository Version | Data Miner XML Schema Version | Release Type (Full or Path) | Limitations |
|--|----------------------|-------------------------------|-------------------------------|-----------------------------|--|
| SQL Developer 4.0 RTM Version 4.0.0.13 Build MAIN-13.80 | 12.2.0.13.80 | 12.1.0.1.3 | 12.1.0.1.3 | RTM (12/12/13) | None |
| SQL Developer 4.0.1 RTM Version 4.0.1.14 Build MAIN14.48 | 12.2.0.14.48 | 12.1.0.1.4 | 12.1.0.1.4 | RTM (02/25/14) | Data Miner fixes and some minor enhancements |
| SQL Developer 4.0.2 Version 4.0.2.15.21 Build 15.21 | 12.2.0.15.21 | 12.1.0.1.5 | 12.1.0.1.4 | RTM (05/06/14) | Only critical Data Miner fixes were back ported. There was a change to PL/SQL code, so the repository version is changed. The XMLschema 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. |
| SQL Developer 4.0.3 Version 4.0.3.16 Build MAIN-16.84 | 12.2.0.16.84 | 12.1.0.1.5 | 12.1.0.1.4 | RTM (09/16/14) | 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. |
| SQL Developer 4.1 EA1 Version 4.1.0.17 Build MAIN-17.29 | 12.2.0.17.29 | 12.1.0.2.1 | 12.1.0.2.1 | EA1 (12/09/14) | None |
| SQL Developer 4.1 EA2 Version 4.1.0.18 Build MAIN-18.37 | 12.2.0.18.37 | 12.1.0.2.2 | 12.1.0.2.2 | EA2 (03/09/15) | None |
| SQL Developer 4.1 RTM Version 4.1.0.19 Build MAIN-19.07 | 12.2.0.19.07 | 12.1.0.2.3 | 12.1.0.2.3 | RTM (05/04/15) | Not allowed to be installed on Oracle Database 12.2. |
| SQL Developer 4.1.1 RTM Version 4.1.1.19 Build MAIN-19.59 | 12.2.0.19.59 | 12.1.0.2.3 | 12.1.0.2.3 | RTM (05/29/15) | Patch Release included minor Data Miner UI bugs. No Repository upgrade required from 4.1 to 4.1.1. |

Table A-1 (Cont.) SQL Developer Oracle Data Miner Extension and Oracle Data Miner Repository Compatibility

| SQL Developer | Data Miner Extension | Data Miner Repository Version | Data Miner XML Schema Version | Release Type (Full or Path) | Limitations |
|---|----------------------|-------------------------------|-------------------------------|-----------------------------|---|
| SQL Developer 4.1.2 RTM Version 4.1.2.20 Build MAIN-20.64 | 12.2.0.20.64 | 12.1.0.2.4 | 12.1.0.2.3 | RTM (10/23/15) | No changes in Oracle Data Miner since Oracle Data Miner 4.1.1 Extension Patch Release. |
| SQL Developer 4.1.3 Version 4.1.3.20 Build MAIN-20.78 | 12.2.0.20.78 | 12.1.0.2.5 | 12.1.0.2.3 | RTM (12/22/15) | No changes in Oracle Data Miner since Oracle Data Miner Patch Release 4.1.2.20. |
| SQL Developer 4.1.4 Version 4.1.4.21 Build Build MAIN-21.20 | 12.2.0.21.20 | 12.1.0.2.6 | 12.1.0.2.3 | RTM (8/16) | Bug fixes only. Repository upgrade required. Available only through the Oracle Database Installation and not through Oracle SQL Developer OTN site. |
| SQL Developer 4.1.5 Version 4.1.5.21 Build MAIN-21.78 | 12.2.0.21.78 | 12.1.0.2.6 | 12.1.0.2.3 | RTM (9/16/16) | Bug fixes only. No repository upgrade required from Oracle Data Miner 4.1.4. But repository upgrade is required from Oracle Data Miner 4.1.3. |
| SQL Developer 4.2 EA1 Version 14.2.0.16.260 Build 16.260.1303 | 12.2.1.16.260.1303 | 12.2.0.0.0 | 12.2.0.0.0 | Released (9/17/16) | Supports upcoming Oracle Database main release features. Repository upgrade required. |
| SQL Developer 4.2 EA2 Version 4.2.0.16.356 Build 16.356.1154 | 12.2.1.16.356.1154 | 12.2.0.0.1 | 12.2.0.0.1 | Released (12/21/16) | Includes improved Data Mining Workflow Status Polling performance, and bug fixes pertaining to Oracle Data Miner 4.2 features. |
| SQL Developer 4.2 RTM Version 4.2.0.17.089 Build 17.089.1709 | 12.2.1.17.089.1709 | 12.2.0.0.2 | 12.2.0.0.1 | RTM (4/12/17) | Supports Oracle Database 12.2 features. |
| SQL Developer 17.2 RTM Version 17.2.0.178 Build 178.1847 | 17.2.0.178.1847 | 12.2.0.0.2 | 12.2.0.0.1 | RTM (7/11) | No migration required from prior release. Contains fix for demo data installation script. |
| SQL Developer 17.3.1 RTM Version 17.3.1.279 Build 279.0537 | 17.3.1.279.0537 | 12.2.0.0.3 | 12.2.0.0.1 | Released (10/17) | Contains fix for querying Database product version for 18.x. Requires Repository upgrade. |

Table A-1 (Cont.) SQL Developer Oracle Data Miner Extension and Oracle Data Miner Repository Compatibility

| SQL Developer | Data Miner Extension | Data Miner Repository Version | Data Miner XML Schema Version | Release Type (Full or Path) | Limitations |
|---|----------------------|-------------------------------|-------------------------------|-----------------------------|--|
| SQL Developer 17.4.0 RTM 17.4.0.355 Build 355.2349 | 17.4.0.355.2349 | 12.2.0.0.3 | 12.2.0.0.1 | Released (12/17) | No changes from SQL Developer 17.3 release. |
| Data Miner extension patch on 17.4.0 | 17.4.1.039.1834 | 12.2.0.0.3 | 12.2.0.0.1 | Released (2/18) | Bug fixes (Data Miner 17.4 patch) |
| SQL Developer 18.1.0 RTM Version 18.1.0.095 Build 095.1630 | 18.1.0.095.1630 | 12.2.0.0.3 | 12.2.0.0.1 | Released (4/18) | Includes all fixes from Patch 17.4. |
| SQL Developer 18.2.0 RTM Version 18.2.0.183 Build 183.1748 | 18.2.0.183.1748 | 12.2.0.0.4 | 12.2.0.0.1 | Released (7/18) | Contains fix for bug 26999065. |
| SQL Developer 18.3.0 RTM Version 18.3.0.xxx Build xxxx | 18.3.0.xxx.xxxx | 12.2.0.0.5 | 12.2.0.0.1 | Released (10/18) | Contains fixes for bugs SR 3-18111559321, SR 3-17867936451, SR 3-17977197451, SR 3-18247171961 |

Index

A

AL32UTF8 character set, [2-2](#)
ASM, [5-7](#), [5-8](#)
ASSM, [2-3](#), [5-8](#), [5-9](#)

C

cloud database connection, [2-4](#)

D

data
 access to, [4-9](#)
 multi-byte, [2-2](#)
 sample, [3-5](#)
Data Miner
 administration scripts, [5-1](#)
 architecture, [1-1](#)
 big data, [1-1](#)
 installation, [3-1](#)
 resource plan for, [6-2](#)
Data Miner repository
 See repository
Data Miner users
 See users
DBMS_DATA_MINING, [1-4](#)
DBMS_DATA_MINING_TRANSFORM, [1-4](#)
DBMS_PREDICTIVE_ANALYTICS, [1-4](#)
DM\$ tables, [4-2](#)

F

full backup and restore, [5-3](#)

I

installation, [1-2](#)
 Data Miner, [3-1](#)
 JSON parser and Data Guide, [3-8](#)
 Oracle R Enterprise, [1-2](#)
 requirements, [2-2](#)
 SQL Developer, [3-3](#)
internal tables, [4-2](#)

J

job, [7-17](#)
JSON, [1-1](#), [3-8](#)

L

Linux, [2-3](#)
loadjsonschemagen.sql, [3-9](#)
locks
 session, [5-7-5-9](#)
 workflows, [4-6](#), [4-7](#)

M

managing system resources
 for model builds, [6-3](#)
 for parallel processing, [6-5](#)
 for workflow execution, [6-4](#)
 for workloads, [6-2](#)
Microsoft Windows, [2-4](#)
migratebinaryodmr.sql, [5-7](#)
migrateodmr.sql, [5-7](#)

O

objects, [4-2](#)
ODMR\$ tables, [4-2](#)
ODMR\$WORKFLOWS_BACKUP able, [5-3](#)
ODMRSYS, [5-2](#)
ODMRUSER role, [5-7-5-9](#)
Oracle Advanced Analytics, [1-3](#), [2-2](#)
Oracle Automatic Segment Space Management
 See ASM
Oracle Data Mining, [1-2](#)
 APIs, [1-4](#)
 data dictionary views, [1-6](#)
 PL/SQL packages, [1-4](#)
 SQL scoring functions, [1-5](#)
Oracle Data Pump, [5-3](#)
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 RMAN, [5-3](#)
 Oracle Scheduler, [1-2](#)
 Oracle Solaris, [2-4](#)
 Oracle Text, [1-2](#), [2-2](#)
 Oracle Text Knowledge Base, [2-2](#)

P

parallel query, [6-5](#)
 PARALLEL_DEGREE_POLICY=AUTO), [6-5](#)
 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, [1-5](#)
 PROJECT_DELETE
 PL/SQL API, [8-3](#)
 proxy users, [4-3](#)

R

repository, [1-2](#)
 about, [1-1](#), [1-2](#)
 access rights to, [4-7](#)
 api, [8-1](#)
 backing up and restoring, [5-2](#)
 binary storage, [5-8](#), [5-9](#)
 determining the status and version of, [5-2](#)
 dropping, [5-9](#)
 installation of, [3-1](#)
 installing using the GUI, [3-6](#)
 installing with a script, [3-3](#)
 object-relational storage, [5-8](#), [5-9](#)
 objects in, [4-1](#)
 views, [8-9](#)
 repository view
 ODMR_USER_PROJECT_WORKFLOW,
 [8-10](#)
 ODMR_USER_WF_CLAS_TEST_RESULTS,
 [8-16](#)
 ODMR_USER_WF_REGR_TEST_RESULTS,
 [8-18](#)
 ODMR_USER_WF_TEST_RESULTS, [8-19](#)
 ODMR_USER_WORKFLOW_ALL, [8-11](#)
 ODMR_USER_WORKFLOW_LOG, [8-12](#)
 ODMR_USER_WORKFLOW_MODELS, [8-15](#)
 ODMR_USER_WORKFLOW_NODES, [8-14](#)

resource plan, [6-2](#)
 RMAN, [6-1](#)

S

sample data, [3-5](#)
 sample schemas, [2-2](#)
 scheduler, [7-17](#)
 Scheduler, [6-4](#), [6-6](#)
 scoring, [1-5](#)
 snippets, [1-4](#), [1-5](#)
 SQL Data Mining functions, [1-5](#)
 SQL Developer
 about, [1-1](#)
 installing, [3-3](#)
 SQL script generation, [7-1](#)
 use case, [7-1](#)

T

tables and views, [4-2](#)

U

upgradeRepoFromORtoBinary.sql, [5-7](#)
 upgradeRepoWithNewTableSpace.sql, [5-7](#)
 user's schema, [4-2](#)
 users, [4-1](#)
 Data Miner, [4-1](#)
 objects in user schemas, [4-1](#)
 proxy, [4-3](#)
 shared access to repository, [4-6](#)

W

WF_DELETE
 PL/SQL API, [8-8](#)
 WF_EXPORT
 PL/SQL API, [8-9](#)
 WF_IMPORT
 PL/SQL API, [8-8](#)
 WF_RENAME
 PL/SQL API, [8-7](#)
 WF_STOP
 PL/SQL API, [8-7](#)
 workflows, [1-1](#), [1-2](#), [4-1](#)
 backing up, [5-3](#)
 restoring from backup, [5-4](#), [5-5](#)
 scheduling, [6-4](#)

X

XML
 binary storage, [2-2](#), [2-3](#), [5-8](#), [5-9](#)

XML (*continued*)

object-relational storage, [2-2](#), [2-3](#), [5-8](#), [5-9](#)

XML DB, [1-2](#), [2-2](#), [2-3](#)