Oracle® Data Mining

Administrator's Guide

10*g* Release 1 (10.1)

December 2003

Part No. B10697-01

1 Introduction

This document describes how to install the Oracle Data Mining (ODM) software and how to perform other administrative functions common to all ODM administration. Platform-specific information is contained in a README file for each platform.

1.1 Intended Audience

This administrator's guide is intended for anyone planning to install and run Oracle Data Mining — either a database administrator or a system administrator.

1.2 Structure

This guide is organized as follows:

- **Section 2, "Overview":** Briefly describes Oracle Data Mining 10*g* Release 1 (10.1) .
- **Section 3, "Oracle Data Mining Installation":** Describes the generic installation steps and upgrade information. Platform-specific information is in the platform-specific README file.
- **Chapter 4, "Database Configuration Issues":** Describes the database configuration issues that can affect ODM performance.
- **Section 5, "Oracle Data Mining Administration":** Describes topics of interest to administrators, including improving Oracle Data Mining performance, detecting errors, etc.

- **Section 6, "ODM Native Model Export and Import":** Describes using the PL/SQL interface to perform Model Export and Import, including requirements and restrictions.
- **Section 7, "Documentation Accessibility":** Describes Oracle documentation accessibility standards.

1.3 Where to Find Further Information

The documentation set for Oracle Data Mining is part of the *Oracle10g Database Documentation Library*; the ODM document set consists of the following documents:

- *Oracle Data Mining Administrator's Guide*, 10*g* Release 1 (10.1) (this document). Includes generic installation information.
- For platform-specific installation information, see the platform-specific README files.
- *Oracle Database 10g Installation Guide* for your platform.
- *Oracle Data Mining Concepts*, 10*g* Release 1 (10.1) .
- *Oracle Data Mining Application Developer's Guide,* 10*g* Release 1 (10.1) .

For detailed information about the ODM Java API, see the ODM Javadoc in the directory `$ORACLE_HOME/dm/doc/odmjdoc.zip` (for Windows, `%ORACLE_HOME%\dm\doc\odmjdoc.zip`) on any system where ODM is installed. To prepare the Javadoc for user access, unzip this file so that users can display it in a browser.

1.3.1 Related Manuals

For more information about the Oracle database, see:

- *Oracle Database Administrator's Guide*
- README for your platform
- *Oracle Universal Installer Concepts Guide*
- *Oracle Database Migration*
- *PL/SQL Packages and Types Reference*

1.4 Conventions

In this manual, Windows refers to the Windows 2000 and Windows XP operating systems.

The SQL interface to Oracle is referred to as SQL. This interface is the Oracle implementation of the SQL standard ANSI X3.135-1992, ISO 9075:1992, commonly referred to as the ANSI/ISO SQL standard or SQL92.

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

2 Overview

Oracle Data Mining (ODM) embeds data mining within the Oracle database. The data never leaves the database — the data, data preparation, model building, and model scoring results all remain in the database. This enables Oracle to provide an infrastructure for application developers to integrate data mining seamlessly with database applications.

Data mining functions such as model building, testing, and scoring are provided via a Java API and a PL/SQL API.

Oracle Data Mining supports the following features:

- For classification: Naive Bayes, Adaptive Bayes Networks, and Support Vector Machines
- For regression: Support Vector Machines
- For clustering: *k*-means and O-Cluster
- For association: A Priori
- For attribute importance: Minimum Description Length (MDL)
- For feature extraction: Non-Negative Matrix Factorization
- For unstructured data mining: Text Mining
- For sequence matching and annotation: BLAST

For detailed information about the classes that constitute the ODM Java API, see the Javadoc descriptions of classes.

For detailed information about the subprograms and functions that constitute the ODM PL/SQL API, see the *PL/SQL Packages and Types Reference*.

Oracle Data Mining 10*g* Release 1 (10.1) has many new features. For details, see *Oracle Data Mining Concepts*.

ODM 10*g* Release 1 (10.1) runs on Real Application Clusters (see Section 3.4).

Oracle 10*g* Release 1 (10.1) supports multi-user configuration.

3 Oracle Data Mining Installation

This section specifies generic ODM requirements and provides a description of the generic installation steps.

3.1 ODM Requirements

ODM is an option to Oracle Enterprise Edition. All the software that ODM requires is included in the Enterprise Edition.

3.2 Installation Steps

This document provides the generic instructions for installing Oracle Data Mining.

Before you install ODM, confirm that your system satisfies the software and hardware requirements for Oracle Enterprise Edition, as described in the README for your platform. You should also ensure that your system contains enough space for the tables that you plan to use during data mining.

There are three common cases for installing ODM:

- Oracle and ODM are not installed on your system (Section 3.2.1).
- Oracle9*i* release 1 (or earlier) is installed on your system (Section 3.2.2)
- Oracle9*i* release 2 is installed on your system (Section 3.2.2)

To install ODM on an Oracle10*g* Real Application Cluster, see Section 3.4.

3.2.1 No Database Installed

If this is a first-time installation of ODM on a system where the current release of Oracle is not installed, there are two basic ways to install the Oracle Enterprise Edition:

1. Create a database with the starter database (Section 3.2.1.1).
2. Create a customized database, that is, do not use the starter database (Section 3.2.1.2).

3.2.1.1 ODM Installation with a Starter Database Oracle provides a starter database that automatically includes features that result in a highly effective database that is easy to manage.

Follow these steps to install Oracle and ODM:

1. Start Oracle Universal Installer (OUI). For details, see the *Oracle Universal Installer Concepts Guide*. The OUI starts with a welcome screen and prompts you through a series of steps. Follow the instructions, and

see the release notes for late-breaking information that may affect the installation steps or your choices. After you have specified the source and destination, continue with the following steps in OUI:

2. Installation Types: Select the Enterprise Edition.
3. Database Configuration: Select a configuration. If you are not sure which configuration to choose, select "Create a starter database" and select "General-purpose database", or see Section 3.2.1.2 for information about installing ODM with a customized database.
4. Database Configuration Options: Provide a global database name and a SID, a database character set, and indicate whether you would like to install example schemas.
5. Database File Storage Options: Select File System or Automated Storage Management or Raw Devices.
6. Database File Location: If you choose File System, specify the file location.
7. Specify backup and recovery options.
8. Specify database schema passwords.
9. Select Database Management option.
10. Summary: Presents a list of settings and products to be installed. Click Install.

After successful installation, all ODM software is located in the `$ORACLE_HOME/dm` (for Windows, `%ORACLE_HOME%\dm`) directory. Perform the following post-installation steps:

1. You may want to "unlock" the `DMSYS` account and change the default passwords.
2. Create a tablespace to be used by data mining users.
3. You need at least one user account for data mining, with the appropriate privileges set for that user.
   - To create a user account, go to`$ORACLE_HOME/dm/admin` (for Windows, `%ORACLE_HOME%\dm\admin`) and run `odmuser.sql`.
   - If you already have a user account for data mining, make sure that the user has the privileges specified in the SQL script `odmuser.sql`.
4. Edit your `init.ora` file to set a value for `utl_file_dir` initialization parameter. The value should be the path name of a directory that the database can write to.

3.2.1.2 ODM Installation with a Customized Database

Installing and creating a customized database involves more steps than creating a starter database, but gives you full control to specify database components that you wish to install.

These are the major steps required to install ODM without using a starter database:

1. Install Oracle Enterprise Edition and create a customized database. See Section 3.2.3 for information about recommended database parameter settings for running ODM and Section 4 about tuning your database for improved performance.
2. Run the Oracle Database Configuration Assistant (DBCA) utility to install the ODM option; DBCA is described in the *Oracle Database Administrator's Guide*. You will have the option of selecting the ODM Scoring Engine

After successful installation, all ODM software is located in the `$ORACLE_HOME/dm` (for Windows, `%ORACLE_HOME%\dm`) directory.

In order to run ODM sample programs, certain data sets need to be loaded into the ODM user account. The loading script is at `$ORACLE_HOME/dm/admin/dmuserld.sql` (for Windows, `%ORACLE_HOME%\dm\admin\dmuserld.sql`).

3.2.2 Upgrade from Oracle9*i* Releases

If Oracle9*i* Release 1 (9.0.1) or Release 2 (9.2.0) with the ODM option is installed on your system, you can choose to upgrade your system to the current release. ODM is upgraded as part of the database upgrade process.

For detailed information about upgrading the database, see *Oracle Database Migration*. For information about upgrading ODM, see Section 3.6.

3.2.3 Database Initialization Parameters for Oracle Data Mining

The default values of initialization parameters in an Oracle starter database are generally sufficient for running ODM.

Make sure that `job_queue_processes` is set to a value appropriate for your application (a minimum of 2).

The parameter `utl_file_dir` must be set to a directory path specific to your site.

3.3 Verifying ODM Installation

Oracle10*g* Data Mining is an option to the Oracle10*g* Enterprise Edition. If ODM is part of your installation, the following query should return a value of `TRUE`:

```
SELECT value
FROM v$option
WHERE parameter = 'Oracle Data Mining';
```

This query is usually run by the DBA logged in as dba.

3.4 ODM Installation on a Real Application Cluster

ODM installation on a Real Application Cluster (RAC) is similar to ODM installation on a non-RAC system. If you use Oracle Universal Installer to create the preconfigured database on RAC, ODM will be installed in this database just as it is in a non-RAC environment.

If you choose to create a customized database on your Real Application Cluster (RAC) and install ODM there, we recommend that you configure the ODM tablespace with a raw device partition of at least 250 MB.

3.5 Data Mining Scoring Engine Installation

Data Mining Scoring Engine is a custom installation option for Oracle Data Mining. Select this option to install the ODM Scoring Engine as an alternative to installing Oracle Data Mining.

For more information about the Oracle Data Mining Scoring Engine, see *Oracle Data Mining Concepts*.

3.6 Upgrading ODM

ODM upgrade is part of the Oracle RDBMS 9.2.0 to 10.1.0 upgrade process. When the database server upgrade completes, ODM is upgraded to the 10.1.0 release level.

In order to upgrade ODM 9.2.0 to ODM 10.1 release, you must upgrade your RDBMS to the latest RDBMS 9.2.0.4 patch set release level before starting the migration from 9.2 to 10.1. ODM is part of the RDBMS 9.2.0.4 patch set release. For detailed information about upgrading an Oracle database, see the *Oracle Database Migration* manual.

3.6.1 ODM Schema Object Upgrade

There are major schema changes between ODM 9.2 and the current release. These changes are required to fully support the ODM multi-user environment and to implement Oracle Advanced Security features.

In ODM 9.2, there were two ODM-required database schemas, namely, ODM and ODM_MTR. In the current release, these two schemas have been upgraded to DMSYS and the DM user schema (the former ODM schema). The DMSYS schema is the ODM repository, which contains data mining metadata. ODM schema becomes the DM user schema that holds user input and output/result data sets. Customers can choose to either use the upgraded ODM schema or create one or more data mining user schema(s) to perform data mining activities.

When you upgrade to the current release, the existing ODM 9.2 data mining models, settings, and results are upgraded to the current release format. Customers can continue to conduct various data mining activities using objects upgraded from the 9.2 release. There are schema definition changes in the current release schema.

New objects created in the ODM 10.1 environment are subject to a naming restriction, that is, names of objects must be 25 bytes or less. This restriction applies across DM user database schemas. However, after upgrading, 9.2 object names (models, settings, and results) are retained in the current release environment. It is recommended that users follow the new ODM naming convention when creating objects in the future.

In the 9.2 release, all mining activities are conducted through the ODM schema (with definer's rights). In the current release, data mining activities are performed in the DM user schema (with invoker's rights). In an upgraded ODM environment, the ODM schema has been upgraded from a definer's schema to an invoker's schema.

If necessary, ODM schema objects can be downgraded to the 9.2.0.4 final patch set release.

3.6.2 Category Data Type in 9.2 and in the Current Release

In ODM 9.2, we did not store category data type in the `dm_category_matrix_entry` table. In the current release, we do store data type. In migrating from 9.2 to the current release, this results in all categories restored having a string data type, no matter what the actual data type.

3.7 Sample Programs for Oracle Data Mining

The directory `$ORACLE_HOME/dm/demo/sample` (on UNIX) or `%ORACLE_HOME%\dm\demo\sample` (on Windows) contains sample programs for ODM. This directory contains the following subdirectories:

- `java` — contains ODM sample programs illustrating the Java API. Property-based ODM Java sample programs are removed from the product shipment in 10g. They are downloadable from OTN.
- `plsql` — contains ODM sample programs illustrating the use of the ODM PL/SQL packages `DBMS_DATA_MINING` and `DBMS_DATA_MINING_TRANSFORMS` (in the *PL/SQL Packages and Types Reference*.

The directory `plsql` contains a subdirectory `utl`; contains sample programs illustrating how to export and import ODM models.

The data used by all the sample programs is in `$ORACLE_HOME/dm/demo/data` on Unix or `%ORACLE_HOME%\dm\demo\data` on Windows. ODM sample data sets need to be loaded into a user schema prior to using the sample programs. Refer to the following scripts for creating Oracle tablespace, user schema, and loading ODM sample data sets:

```
$ORACLE_HOME/dm/admin/odmtbs.sql
$ORACLE_HOME/dm/admin/odmuser.sql
$ORACLE_HOME/dm/admin/dmuserld.sql
```

3.7.1 ODM Sample Programs Using Oracle Common Schema (Sales History)

For 10*g*, ODM Java and PL/SQL sample programs also use datasets shipped with Oracle Common Schema (SH). In order to use the datasets, the Sample schema SH must be installed by a site DBA in the target database.

The following table objects in SH schema are referenced by DM Sample programs:

```
sh.sales
sh.customers
sh.products
sh.supplementary_demographics
sh.countries
```

The following scripts need to be executed by the site DBA. The scripts grant necessary SH access privileges and create related DM objects prior to running DM sample programs that reference SH schema objects:

```
$ORACLE_HOME/dm/admin/dmshgrants.sql
$ORACLE_HOME/dm/admin/dmsh.sql
```

3.8 Downgrading ODM

ODM 10.1 can be downgraded if customers are not satisfied with the results of upgrading ODM 9.2 to 10.1. The downgrade must comply with RDBMS downgrade policy. The initialization parameter `COMPATIBLE` needs to be retained as 9.2.0 in the database during the upgrade process.

Once the RDBMS downgrade process completes, ODM will be downgraded to the latest 9.2.0 patch set release level. The ODM repository schema in the database will be ODM. `ODM_MTR` schema will be retained.

3.9 Deinstalling ODM

You can use the OUI to deinstall ODM.

4 Database Configuration Issues

This section summarizes the database configuration issues that can influence ODM performance, given the respective hardware resource. Many Oracle initialization parameters are tunable via `init`*`SID`*`.ora` file, which is located under `$ORACLE_HOME/dbs` directory. A pre-configured database (`SeedDB`, also referred to as starter database) sets many parameters with default values. ODM users can tune these values based on site-specific circumstances.

For detailed descriptions, refer to Oracle SQL Reference and *Oracle Database Administrator's Guide*.

4.1 Shared Global Area (SGA)

Subject to physical memory capacity, the database System Global Area (SGA) should be set adequately to enhance the database performance. A DBA should determine how much total memory on the system is available for Oracle database to consume (referred to as "available memory"). A certain amount of physical memory on the system needs to be reserved for buffering and process memory consumption.

SGA size consists of the following init parameter settings:

Table 1 Init Parameter Settings for SGA Size

Parameter	Description
`shared_pool_size`	Specifies (in bytes) the size of the shared pool. The shared pool contains shared cursors, stored procedures, control structures, and other structures. The size should be set as 5-10% of the available memory.

Table 1 (Cont.) Init Parameter Settings for SGA Size

Parameter	Description
db_cache_size	The DB_CACHE_SIZE parameter specifies the size of the cache of standard block size buffers, where the standard block size is specified by DB_BLOCK_SIZE. The size should be set as 20- 80% of the available memory.
log_buffer	Specifies the amount of memory (in bytes) when buffering redo entries to a redo log file. Redo log entries contain a record of the changes that have been made to the database block buffers.

v$sgastat records SGA dynamic allocation stats. For details, refer to the *Oracle Administrator's Guide*.

More memory-related tunable parameters are described as below:

Table 2 Tunable Parameters Related to Memory

Parameter	Description
java_pool_size	Specifies the size (in bytes) of the Java pool, from which the Java memory manager allocates most Java state during runtime execution.
large_pool_size	Specifies the size (in bytes) of the large pool allocation heap. The large pool allocation heap is used in shared server systems for session memory, by parallel execution for message buffers, and by backup processes for disk I/O buffers.
sort_area_size	Specifies in bytes the maximum amount of memory Oracle will use for a sort.
hash_area_size	Specifies the maximum amount of memory, in bytes, to be used for hash joins.
pga_aggregate_size	Introduced in 9*i*. The parameter manages runtime memory allocation. It replaces hash_area_size, sort_area_size, create_bitmap_area_size, and bitmap_merge_area_size parameters. Recommended to be set as 20 -80% of the available memory.

4.2 Parallel Queries (PQ)

The following PQ parameters are tunable:

Table 3 Tunable PQ Parameters

Parameter	Description
`parallel_max_servers`	Maximum parallel server processes (setting value is subject to CPU number on the host).
`parallel_min_servers`	Minimum parallel server processes.
`parallel_min_percent`	Operates in conjunction with `parallel_max_servers` and `parallel_min_severs`. It sets the minimum percentage of parallel execution processes (of the value of `parallel_max_servers`) required for parallel execution.
`parallel_automatic_tuning`	Setting `parallel_automatic_tuning` to `TRUE` will result in the database configuring itself to support parallel execution (default is `FALSE`).
`parallel_threads_per_cpu`	Describes the number of parallel execution processes or threads that a CPU can handle during parallel execution. If the machine appears to be overloaded, decrease the value of this parameter; if the system is I/O bound, increase the value.

Most PQ settings are subject to the available number of CPUs on the host. For machines with a single CPU, the parallel execution is limited. ODM algorithms in most cases use default parallel degree setting. The number of CPUs and their capacity largely influences the parallelism.

The v$process view records the status for all slave processes.

4.3 Multi-Threaded Server (MTS)

Multi-Threaded Server configuration enables a large number of user sessions to share the same server process; workload is distributed via the dispatcher. In MTS configuration, the User Global Area is part of the SGA; hence a larger SGA configuration is recommended. The actual degree of increase is subject to the values of MTS-related init parameters.

MTS-related init parameters are:

Table 4 Init Parameters Related to MTS

MTS Parameter	Description	Recommended Setting
`dispatchers`	Specifies dispatcher processes in the shared server architecture.	2-10
`max_dispatchers`	Specifies the maximum number of dispatcher processes that can run simultaneously.	10
`shared_servers`	Specifies the number of shared server processes created when an instance is started up.	2-10
`max_shared_servers`	Specifies the maximum number of shared server processes that can run simultaneously.	10

Check `v$dispatcher` and `v$shared_server` for runtime dispatcher and shared server status.

5 Oracle Data Mining Administration

This section contains information of interest to ODM administrators.

For information about administering an Oracle database, see the *Oracle Database Administrator's Guide*.

5.1 Improving ODM Performance

There are two ways to improve performance: By enabling parallelism and by compiling clustering procedures into native code in shared libraries.

- To improve ODM performance, enable parallelism by setting the database initialization parameters `PARALLEL_MAX_SERVERS` and `PARALLEL_MIN_SERVERS` based on the characteristics of your system, particularly the CPU number of your system.
- You can speed up clustering package (`dmcuh`, `dmcub`, `dmkmh`, `dmkmb`, `dmoch`, `dmocb`) procedures by compiling them into native code residing in shared libraries. Oracle translates the procedures into C code. You then compile with your usual C compiler and link into the Oracle process. For details on how to compile PL/SQL procedures into native code, see the *PL/SQL User's Guide and Reference*.

Under certain circumstances, ODM models can be of considerable size. This may lead to large memory requirements at scoring time. By default, ODM scoring runs in parallel. However, if memory resources are insufficient, we recommend disabling parallelism for certain types of models to alleviate memory requirements. The models to which this applies are:

- SVM models with non-linear kernels
- SVM models with linear kernels for data with very large number of dimensions and target classes
- *k*-Means clustering models (PL/SQL API only) for data with very large number of dimensions and number of clusters.

Subject to available CPU numbers and capacity, a site DBA may tune Oracle parallelism-related initialization parameters to adjust parallelism level on a host when using ODM-specific algorithms. At session level, the parallel DML can be enabled or disabled. Setting appropriate values for the following init parameters and others (details refer to Oracle SQL Reference documentation).

```
parallel_automatic_tuning
parallel_max_servers
parallel_min_percent
parallel_min_servers
parallel_threads_per_cpu
```

5.2 Changing DMSYS Password

Change the `DMSYS` default password after installation completes. You change the password just as you change any other database password.

5.3 ODM Configuration Parameters

The following ODM configuration parameters reside in the `DM$CONFIGURATION` table. These parameters may require modification for your environment.

ABN_ALG_MAX_ATTRIBUTES

The maximum number of predictors is a feature selection mechanism that can provide a substantial performance improvement, especially in the instance of wide training tables. Note that the predictors are rank ordered with respect to an MDL measure of their correlation to the target which is a greedy measure of their likelihood of being incorporated into the model. The actual number of predictors will be the minimum of the parameter value and the number of active predictors in the model.

If the value is less than the number of active predictors in the model, the predictors are chosen in accordance with their MDL rank. The default is 25.

The number of predictors in the baseline Naive Bayes model is restricted by another mechanism: `numberOfPredictorsInNBModel`, which can be more or less than `MaximumPredictors`. Valid range: 1 - infinity

ABN_ALG_SETTING_NB_PRED
The number of predictors in the NB model. The actual number of predictors will be the minimum of the parameter value and the number of active predictors in the model. If the value is less than the number of active predictors in the model, the predictors are chosen in accordance with their MDL rank. Default is 10. This setting is ignored if the model type is neither `NaiveBayesBuild` nor `MultiFeatureBuild`. Valid range: 1 - infinity

ABN_ALG_SETTING_NF_DEPTH
Data type is `int`; default is 10. Specifies the maximum depth of any Network Feature for ABN setting.

ABN_ALG_SETTING_NUM_NF
Data type is `int`; default is 1. Specifies the maximum number of Network Features for ABN setting.

ABN_ALG_SETTING_NUM_PRUNED_NF
Data type is int; default is 0. Specifies maximum number of consecutive pruned Network Features for ABN setting

AI_BUILD_SEQ_PER_PARTITION
Data type is `int`; default is 50000. Specifies

AUTO_BIN_CATEGORICAL_NUM
Data type is `int`; default is 5. Specifies the number of bins used by automated binning for categorical attributes. This value should be >=2.

AUTO_BIN_CATEGORICAL_OTHER
Data type is `STRING`; default is `OTHER`. Specifies the name of the "Other" bin generated during Top-*n* categorical binning.

AUTO_BIN_CL_NUMERICAL_NUM
Data type is `int`; default is 100. Specifies the maximum number of bins allowed for numerical attributes for clustering. Useful values are between 2 and 100. This parameter is used in conjunction with `CL_ALG_SETTING_KM_BIN_FACTOR` and `CL_ALG_SETTING_OC_BIN_FACTOR`.

AUTO_BIN_NUMERICAL_NUM
Data type is `int`; default is 5. Specifies the number of bins used by automated binning for numerical attributes. This value should be >= 2.

CL_ALG_SETTING_CHI2_LOW
Data type is NUMBER; default is 1.353. Controls the level of statistical significance for O-Cluster to determine if more data is necessary to refine a model.

CL_ALG_SETTING_KM_BIN_FACTOR
Data type is NUMBER; default is 2. Factor used in automatic bin number computation for the *k*-means algorithm. Increasing this value will increase resolution by increasing the number of bins. However, the number of bins is also capped by AUTO_BIN_CL_NUMERICAL_NUM.

CL_ALG_SETTING_KM_BUFFER
Data type is int; default is 10000. Number of rows used by the in-memory buffer used by *k*-means. For an installation with limited memory, this number should be smaller than the default data size. Summarization is activated for data sets larger than the buffer size.

CL_ALG_SETTING_KM_FACTOR
Data type is NUMBER; default is 20. Controls the number of points produced by data summarization for *k*-means. The larger the value, the more points. The formula for the number of points is:

Number of Points = CL_ALG_SETTING_KM_FACTOR *

* Num_Clusters

where Num_Attributes is the number of attributes and Num_Clusters is the number of clusters.

The number of points must be <= 1000. This parameter can be any positive value; however, a small number of summarization points can produce poor accuracy.

CL_ALG_SETTING_MIN_CHI2_POINTS
Data type is int; default is 10. Controls the minimum number of rows required by O-Cluster to find a cluster. For data tables with a very small number of rows, this number should be set to a value between 2 and 10.

CL_ALG_SETTING_OC_BIN_FACTOR
Data type is NUMBER; default is 0.9. Factor used in automatic bin number computation for the O-Cluster algorithm. Increasing this value will increase the number of bins. However, increasing the number of bins may have a negative effect on the statistical significance of the model.

CL_ALG_SETTING_OC_BUFFER
Data type is int; default is 50000. Specifies the number of rows used by the in-memory buffer used by O-Cluster. For an installation with limited memory, this number should be smaller than the default size.

CL_ALG_SETTING_TREE_GROWTH
Data type is `int`; default is 1. Must be 1 or 2. 1 specifies that *k*-means is a balanced tree; 2 specifies that *k*-means is an unbalanced tree.

CLASSIFICATION_APPLY_SEQ_PER_PARTITION
Data type is `int`; default is 50000. Specifies the maximum number of unique sequence IDs per partition used by clustering apply.

CLASSIFICATION_BUILD_SEQ_PER_PARTITION
Data type is `int`; default is 50000. Keeps the computations constrained to memory-sized chunks and determines the size of the random sample used for MDL computations (scoring within build). There is no maximum value; this value should not be smaller than 1000.

CLUSTERING_APPLY_SEQ_PER_PARTITION
Data type is `int`; default is 50000. Constrains the scoring to memory-sized chunks of the data and loop through such chunks. The value for this parameter depends on the sorting area (SA) and the number of clusters. The larger the SA, the larger this parameter can be. A rough formula for this parameter is

`CLUSTERING_APPLY_SEQ_PER_PARTITION` = `SA`/(100*`Num_Clusters`)

where `SA` is the size of the sorting area and `Num_Clusters` is the number of clusters.

LOG_LEVEL
Data type is `int`; default is 2. Must be 0, 1, 2, or 3. Specifies the type of messages written to the `LOG_FILE`. 0 means no logging; 1 means write Internal Error, Error, and Warning messages to `LOG_FILE`; 2 means write all messages for 1 plus Notifications; 3 means write all messages for 2 plus trace information.

ODM_CLIENT_TRACE
Data type is `int`; default is 0. Must be 0, 1, 2, or 3. Enables trace for the ODM client. 0 indicates no trace; 1 indicates low; 2 indicates moderate; 3 indicates high.

ODM_SERVER_JAVA_TRACE
Data type is `int`; default is 0. Must be 0, 1, 2, or 3. Enables trace for the ODM client. 0 indicates no trace; 1 indicates low; 2 indicates moderate; 3 indicates high.

ODM_SERVER_SQL_TRACE
Data type is `int`; default is 0. Must be 0, 1, 2, or 3. Enables trace for the ODM client. 0 indicates no trace; 1 indicates low; 2 indicates moderate; 3 indicates high.

5.4 Need for Compatible Character Sets

All connections made to an ODM server must be based on databases with compatible character sets. Otherwise, string length tests conducted in the JVM may not recognize differences, allowing data to pass to the database, which could result in server-side failures.

5.5 ODM Errors

When you encounter an error during the execution of a method, the ODM server outputs two kinds of error messages:

- Error messages prefixed with an ORA-20xxx number. Consult the `odm_error.txt` in your installation for an explanation of the ORA-20xxx error.
- Error messages with an ORA-40101 error. These are errors caused during invocation of a mining operation. You may study the error stack that follows this message and take remedial action where possible. Examples are inadequate temporary segment or rollback segments. Consult Oracle Support if you are not able to identify the problem from the error stack.

All error messages are written to the `dm_message.log` file located in the directory specified by the `init` parameter `UTL_FILE_DIR`. The value of this parameter varies by site.

5.6 Deployment Scenarios

Multiple ODM users can share the same database instance in using ODM 10*g*.

There are four SQL scripts provided for a site DBA to create a data mining user account and load demo data sets:

1. `$ORACLE_HOME/dm/admin/odmtbs.sql` — creates a user tablespace
2. `$ORACLE_HOME/dm/admin/odmuser.sql` — creates a user account
3. `$ORACLE_HOME/dm/admin/dmuserld.sql` — loads DM demo data sets to an existing user schema
4. `$ORACLE_HOME/dm/admin/dmindcrt.sql` — creates a Text index for the text demo table

The site DBA needs to understand that certain input parameters are required in using these scripts. Check the Note section in these scripts for details.

6 ODM Native Model Export and Import

Oracle Data Mining supports import and export of models created using the PL/SQL interface. Export and import are accomplished using Oracle Data Pump, described in *Oracle 10g Database Utilities, Part 1*. Export and import are usually performed by a data mining administrator or a DBA.

6.1 Restrictions on ODM Model Export and Import

You can export and import data mining models subject to the following restrictions:

- You can export and import models created using the PL/SQL interface to ODM; you cannot export or import models created using the ODM Java API.
- Remote export/import via database link is not yet supported.

6.2 Prerequisites for ODM Model Export and Import

For model export,

- Directory objects must be created to map the location of the output dump file and log file. The operator must be granted read and write privileges to the directory object.
- The destination must be an Oracle database with either the Oracle Data Mining option or the Oracle Data Mining Scoring Engine option installed.

The Oracle Data Pump Export Utility (`expdp`) is used for database and schema export.

For model import,

- A valid directory object mapped to the location of the dump files must exist; the operator must have read and write privileges.
- The destination database must have either the Oracle Data Mining option or the Oracle Data Mining Scoring Engine option installed.
- Dump files must be created using either the Oracle Data Pump Export Utility (`expdp`) or `DBMS_DATA_MINING.export_model()`.

The Oracle Data Pump Import Utility (`impdp`) is used for database and schema import.

Native model export and import is based on Data Pump technology. Readers of this chapter are strongly urged to read the *Oracle10g Database Utilities* manual first, in order to be familiar with this technology.

6.3 Using Native Model Export and Import

Data mining models can be moved between Oracle databases or schemas. For example, data mining specialists may build and test data mining models in a data mining lab. After the models are built and tested in the lab, the chosen models may be deployed to another instance of Data Mining Server, for instance, a scoring engine, to be used by applications. Because the data mining lab and the scoring engine usually do not share the same database, the model must be exported from the lab and then imported to the scoring engine.

Model export and import can be a routine procedure. As new data are accumulated, data mining specialists will build and test more new models, and newer and better models will be loaded onto the scoring engine on a regular basis.

DBAs may want to back up and restore models in their routine database maintenance.

Native mode export and import are supported at three different levels, as follows:

- When a full database is exported using the Oracle Data Pump Export Utility (`expdp`), all data mining models in the database are exported. When a database is imported using the Oracle Data Pump Import Utility (`impdp`), all data mining models in the dump are restored to the destination database.
- When a schema is exported using the Oracle Data Pump Export Utility (`expdp`), all data mining models in the schema will be exported. When a schema is imported using the Oracle Data Pump Import Utility (`impdp`), all data mining models in the schema are restored to the destination schema.
- An ODM user can export one or more specific models in a schema using the `DBMS_DATA_MINING.export_model` procedure. An ODM user can import one or more specific models from a dump file using the `DBMS_DATA_MINING.import_model` procedure.

6.4 ODM Model Export and Import Procedures

This section describes the steps to follow to export and import ODM data mining models.

6.4.1 Choose the Right Tool for Model Export and Import

For more information about import and export, see

- The Oracle Data Pump Export and Oracle Data Pump Import discussions in *Oracle Database Utilities, Part I*.

- The DBMS_DATA_MINING subprograms discussion in *PL/SQL Packages and Types Reference.*

Oracle strongly recommends that you use the new Data Pump Export and Import utilities (expdp and impdp) for database export and import starting with the current database release.

The classic tools exp and imp still work in the current release. However, exp will be de-supported in future releases, while imp is provided only for backward compatibility,

There are two ways to export models:

- Export all models in a user schema or in the entire database
- Export selected models in a user schema

To export all data mining models from a database, run expdp as you would normally do to export the full database.

To export all data mining models in a user schema do one of the following:

- Run expdp as you would normally do to export a schema
- Execute the ODM PL/SQL procedure DBMS_DATA_MINING.export_model with a NULL model filter.

There is a difference between the two operations. When you run expdp to export the schema, all objects in the schema including data mining models will be exported. When you run DBMS_DATA_MINING.export_model with a NULL model filter, only data mining models are exported.

To import data mining models from a dump file, you may choose one of the two ways,

- Run impdp to import all data mining models as well as other database objects
- Run DBMS_DATA_MINING.import_model to import data mining models only, either all models or selected models

The Oracle Data Pump Utility impdp imports all or part of database objects and data, as you choose to import, and all data mining models as well from the dump file set. If you want to import models only, you use the PL/SQL procedure DBMS_DATA_MINING.import_model.

For details about the usage of PL/SQL procedures, DBMS_DATA_MINING.export_model, and DBMS_DATA_MINING.import_model, see *PL/SQL Packages and Types Reference.*

Several examples demonstrating how to use them can be found in the *Oracle Data Mining Application Developer's Guide*.

6.4.2 Directory Objects for ODM Model Export and Import

Directory objects are used to specify file locations required by Oracle Data Pump technology. The SQL statement `CREATE DIRECTORY` creates directory objects.

Directory objects may be created and managed in one of two ways:

- A DBA plans and manages all directory objects. The DBA can control the use of each directory object by granting proper privileges on the directory object to individual users.
- Individual users create and manage directory objects on their own.

Usually a DBA manages directory objects so that control of data security and resource management is centralized.

For non-DBA users, the `CREATE ANY DIRECTORY` privilege is needed to create directory objects. Users can find details of all directory objects available by running the following query in SQL*Plus:

```
SQL> select * from ALL_DIRECTORIES;
```

If the specified directory objects are non-existent or if the operator does not have the proper privileges, export and import jobs will fail and will throw an ORA-39002 error.

For information about `CREATE DIRECTORY`, see the *Oracle SQL Reference*. For information about Oracle Data Pump and directory objects, see *Oracle Database Utilities*.

6.4.3 Privileges Required for ODM Model Export and Import

The two roles `EXP_FULL_DATABASE` and `IMP_FULL_DATABASE` are used to allow privileged users to take full advantage of model export and import utilities.

For example, if user `MARY` wants to import data mining models from a dump file set exported by user `SCOTT` from `SCOTT`'s schema, she has to set the parameter `remap_schema` in `impdp`. In order to run `impdp` successfully, `mary` must have been granted the `IMP_FULL_DATABASE` privileges. If user `MARY` does not have the `IMP_FULL_DATABASE` privileges or the `SYS` role, `impdp` issues an error like the following:

```
ORA-31631: privileges are required
ORA-39122: Unprivileged users may not perform REMAP_SCHEMA
remappings.
```

A similar error occurs if `MARY` runs `DBMS_DATA_MINING.import_model` with a non-null schema remap setting:

```
Error=ORA-40223: data mining model import failed,
```

```
job name=SCOTT_imp_82,
error=ORA-31631: privileges are required
```

6.4.4 Temporary Tables Used with ODM Model Export and Import

Data mining model export and import jobs utilize and manage two temporary tables in the data mining user schema: `DM$P_MODEL_EXPIMP_TEMP` and `DM$P_MODEL_TABKEY_TEMP`. The latter is created after the user runs `DBMS_DATA_MINING.import_model` the first time. Users should not manipulate these tables. If `DM$P_MODEL_EXPIMP_TEMP` grows too large, you may truncate it while there are no active export or import jobs running.

6.4.5 How to Find ODM Models in a Dump File

In order to import selected models from a dump file set, you must find the model names contained in the dump file set. The best way to find the models in a dump file set is to read the original export log. If models are exported successfully, the export job log contains lines that look like this:

```
Connected to: Oracle10g Enterprise Edition Release FINAL RELNO.
with the Partitioning and Data Mining options
FLASHBACK automatically enabled to preserve database integrity.
Starting "GT"."SYS_EXPORT_SCHEMA_01": gt/******** DIRECTORY=dm_dump
DUMPFILE=tmdpex01 LO
GFILE=tmdpex01
Estimate in progress using BLOCKS method...
Processing object type SCHEMA_EXPORT/TABLE/TABLE_DATA
Total estimation using BLOCKS method: 78.56 MB
Processing object type SCHEMA_EXPORT/SE_PRE_SCHEMA_
PROCOBJACT/PROCACT_SCHEMA
>>> . . exported Data Mining Model "GT"."ABN_CLAS_SAMPLE"
>>> . . exported Data Mining Model "GT"."AI_SAMPLE"
>>> . . exported Data Mining Model "GT"."KM_SAMPLE"
>>> . . exported Data Mining Model "GT"."NAIVE_BAYES_SAMPLE"
...
```

Therefore, you should keep export logs handy and close to the dump file sets.

If the export log is missing or is not available for any reason, run `impdp` with the `SQLFILE` option. This operation will generate a SQL file containing DDL commands that recreate all database objects in the dump file set. You can find model names by opening the SQL file in a text editor and searching for keyword `create_model`. The model name should be the first string within the parenthesis after every `create_model` keyword.

7 Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/

7.1 Accessibility of Code Examples in Documentation

JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.