

An Oracle White Paper
September 2009

Oracle Database 11g Workspace Manager Overview

Introduction	2
Workspace Manager	2
Creating An Operational Data Store	3
Isolating A Collection Of Changes to Data	3
Performing “What if” Analysis.....	4
Application Development Features	4
Version-enabling tables.....	5
Workspaces	5
Savepoints	6
History of Changes.....	6
Valid Time Support.....	6
Merging, Refreshing and Rolling Back Workspace Changes	7
Conflict Detection and Resolution	7
Workspace Access Modes.....	8
Workspace Locking.....	8
Workspace Privileges.....	8
Workspace Manager Events	9
Spatial Topology Support.....	9
Management Features	9
Removing Unwanted Versions And Workspaces.....	9
Views To Manage Workspaces.....	10
DDL Operations On Version-Enabled Tables	10
Constraints On Version-Enabled Tables.....	10
Triggers On Version-Enabled Tables.....	10
Import And Export Version-Enabled Tables.....	10
Replication Support.....	11
SQL*Loader for Bulk Loading	11
Materialized View Support	11
Using Virtual Private Database Security	12
Support for Table Synonyms.....	12
System Parameters for Workspace Manager	12
New Features for Oracle Database 11g.....	12
Conclusion	13

Introduction

Business processes and DBA operations often work with multiple versions of data. Three common reasons for multi-versioning are for concurrency, history and what-if scenario creation. Versioning for concurrency means data can be inserted and changed in a workspace while users continue to use the production version of the data. Isolating changes in this fashion, also known as a long transaction, is useful for data validation and long duration projects. Versioning for history allows earlier versions of the data to be kept indefinitely. This is useful for users to go back to see how the database looked with the changes at a particular point in time. Versioning for scenario creation is useful for “what if” analysis and application development testing.

Workspace Manager, a feature of Oracle Database allows application developers and DBAs to manage multiple versions of data in the same database. It uses workspaces as a virtual environment to isolate a collection of changes to production data, keep a history of changes to data and create multiple data scenarios for “what if” analysis. It can save money, time and effort over traditional approaches.

Workspace Manager

Workspace Manager is a feature of the Oracle Database for application developers and DBAs. It is a PL/SQL package that "version-enables" user tables. When a table is version-enabled it is renamed. A view is created on it and given the original table name, which makes the renaming of the table transparent to applications. Instead-of triggers created on the view ensure all DML statements executed on the view are applied to the underlying table. Workspace Manager metadata columns added to the user table allow multiple versions of a row with the same user-defined primary key to exist in the table. Standard principles and methodology for database design and tuning apply.

A workspace logically groups a collection of changes (that is new row versions) and provides session context to ensure a user accesses the appropriate version of a row. Workspaces can be arranged in hierarchies. The top most workspace is called LIVE. It is the default workspace for user activity and the production version of the data.

Workspace Manager only makes a copy of a row if it is changed. This can significantly reduce the hardware, software and time needed to manage multiple versions of the data, as compared to scenarios where data is copied in bulk (tables or schemas) and synchronized. It increases productivity by allowing concurrent access to different versions of the data and changing context

to a different workspace. It also reduces labor by allowing a single point of update and management for all versions of the data while freeing the application developer from writing custom code and creating application specific metadata to keep track of multiple data versions. It also does not require changes to application SQL statements to access version-enabled tables.

The following three case studies illustrate some of the ways in which customers use Workspace Manager today.

Creating An Operational Data Store

An operational data store (ODS) for a major system supports key operational business processes. It aggregates transaction-processing data from multiple legacy applications and provides subject-oriented, integrated, near realtime, detailed data for a number of financial applications and reports. A major requirement for the ODS is to provide current, daily and monthly snapshots of the 500 gigabyte Oracle Database on the existing hardware platform. Unfortunately the current hardware has a one-terabyte storage limit. Other requirements include loading 60 megabytes (120,000 transactions) per hour and enabling applications to access the snapshots without changes to existing SQL statements and queries.

Workspace Manager was chosen because it only versions changed rows, requiring less storage than a table or database copy. Users in a workspace automatically see the correct version of the rows in which they are interested along with the rest of the data in the database, as it existed when the workspace was created or last refreshed with changes from the parent workspace. It is also easy to add a new snapshot as another workspace.

In this implementation, three workspaces are used to provide the required snapshots. Workspaces called `LIVE`, `DAILY` and `MONTHLY` allow users to see the current state of the data as well as the data as it existed at the end of the previous day and month, respectively. The `DAILY` and `MONTHLY` workspaces are refreshed at the appropriate time to make the latest changes in `LIVE` visible in the workspace. This is a fast operation because only version metadata is updated. Subsequent changes to the data in `LIVE` are not visible in the other workspaces until the next refresh. The `CompressWorkspace` API is used to remove obsolete versions. Using Workspace Manager, updates to the ODS happen in near real time and snapshots are refreshed with virtually no latency for better operational decision-making.

Isolating A Collection Of Changes to Data

The City of Edmonton developed a Spatial Land Inventory Management System (SLIM) on Oracle Database that provides a single management environment for its land-based assets. SLIM uses GeoMedia Pro, from Intergraph Corporation with Oracle Spatial to manage location data. It uses GeoMedia Transaction Manager with Workspace Manager to manage current, proposed and historical values of the data. SLIM replaced 49 disparate land applications and 166 databases. Before SLIM, data duplication was common, data was maintained in multiple data formats, the

quality of data was inconsistent, currency of data was often a problem, some required data did not exist and limited historical data was available.

Workspace Manager enables SLIM users to store the current, proposed and historical values for data in the same database. Data maintainers create workspaces to isolate a collection of changes for an indefinite period of time. These changes are merged with current data when they are completed and approved. This allows end users to access current data while data is being updated. It also creates an audit trail for data maintainers and keeps historical and proposed states of the data for business users. The Workspace Manager `GotoDate` API enables users use to see the database as it existed at any point in time.

Performing “What if” Analysis

A major transportation company developed a strategic planning application with Workspace Manager. It is intended to manage projects related to the construction, upgrade and repair of its assets. Major application requirements include the ability to create multiple versions of project data and to specify a valid date and time range for each version of the data. These capabilities allow users to analyze the impact and timing of various project scenarios on capacity, resource utilization, bottlenecks, and schedules. Workspace Manager provides these capabilities by versioning project data and allowing a valid time to be specified for each row version.

For example, consider the impact of a proposed railway line extension on an analysis of expense, revenue and population centers served by rail. During data entry, workspaces are used to create multiple project scenarios from a common data set. Each record that is changed for a scenario is stamped with a valid time to reflect when expenses will be incurred, service provided or revenue realized. During data analysis, a user can set various combinations of valid time and workspace in his session context. Subsequent queries return versions that were created from the workspace and stamped with a valid time that falls within the valid time of the session context. This filters records appropriately to analyze the impact and timing of each scenario.

For instance, consider a scenario that has a valid time beginning in 2004. If the valid time for the session context is set between 2003 and 2004 the query results will include population centers located along the new line. If the valid time for the session context is set between 2002 and 2003 the query results will not.

Workspace Manager also provides a history option for a version-enabled table that time stamps versions with the transaction time. This is the time the data is actually entered. It allows users to go to a date to see the database as it existed as of a particular transaction time.

Application Development Features

Workspace Manager provides a comprehensive PL/SQL API that developers can add to new and existing applications to version-enable tables, work in workspaces, use workspace savepoints,

history, privileges, access modes, and locks, and to detect and resolve conflicts. You can also perform Workspace Manager operations using the Oracle Enterprise Manager interface.

Version-enabling tables

Workspace Manager can version-enable one or more user tables in the database. The unit of versioning is a row. When a table is version-enabled, all rows in the table can support multiple versions of the data. Versioned rows are stored in the same table as the original rows. All inserts updates and deletes (DML) to row versions are done in conventional Oracle short transactions, ensuring integrity of versioned data. The versioning infrastructure is not visible to the users of the database.

Workspace Manager implements these capabilities by renaming the table to *table name_LT*, adding a few columns to the table to store versioning metadata, creating a view on the version-enabled table using the original table name and defining INSTEAD OF triggers on the view for SQL DML operations. If you no longer need a table to be version-enabled, you can disable versioning for the table.

Workspaces

A workspace is a virtual environment, not physical storage. It is used in Oracle Database to maximize concurrency, and logically group and isolate a set of changes (that is, new row versions) to data in a long transaction of indefinite duration. Users in a workspace automatically see the correct version of the rows in which they are interested along with a transactionally consistent view of the rest of the data in the database, as it existed when the workspace was created or last refreshed with changes from the parent workspace.

The default workspace for a session context is called LIVE. It is where users usually see the current production version of the data. There can be a hierarchy of workspaces in the database. LIVE is always the topmost workspace. By default, when a workspace is created, it is created as a child of LIVE. A user sets their session context to a workspace other than LIVE by issuing a GotoWorkspace command from their application or login script. A workspace can be used by one user or shared by many users.

A new version of a row is created when a change is made to the row in a child workspace for the first time. Subsequent changes to the row are applied to the same version until a savepoint is created (see Savepoints below). The new row versions created in a workspace are physically contained in the same version-enabled tables as the original versions of the rows. Changes in a workspace can not be seen outside the workspace until they are explicitly merged with production data or discarded.

Workspace operations include create, goto, refresh, merge, rollback, compress, remove, multi-parent and alter description.

A child workspace can be made to have two or more parent workspaces, in which case it becomes a multiparent workspace. A multiparent workspace can see data from all of its parent workspaces and their ancestor workspaces, and it can be merged with and refreshed from its parent workspaces. It is an easy way to query or make a change that needs to be visible in, two or more workspaces.

Users can navigate back in time within a workspace and select intermediate row versions for read only. Executing `GotoDate` sets session context to a historical time. `GotoSavepoint` sets session context to a particular savepoint. Subsequent `SELECT`s will select for read only the latest row version as of the specified date or savepoint.

Savepoints

Savepoints are the mechanism by which new versions are created. They are points in the workspace to which changes can be rolled back, and to which users can go to see the state of the database as it existed as of a particular milestone.

Savepoints can be created implicitly or explicitly. An implicit savepoint is created in the parent workspace when a child workspace is created. An explicit savepoint is created by a user in response to a milestone like a business event, such as the completion of a design or the end of a business period. If a row is modified after a savepoint is created a new version of that row is created. Subsequent changes are applied to this new version in the workspace until another savepoint is created. Users can compare differences between any two savepoints.

History of Changes

When a table is version-enabled, the history option can be chosen. If this option is enabled, Workspace Manager adds a transaction time timestamp (of either the `TIMESTAMP WITH TIME ZONE` or the `DATE` type) every time the row is changed. The history option provides a history of changes made to each new version created by a savepoint. This allows users in a workspace to go back to any point in time and view the entire database from the perspective of changes made in that workspace.

The history option can either make a copy of the row version each time a change is made to it (`without_overwrite` option) or over-write values in the row version with the most recent changes and the current timestamp (`with_overwrite` option). Specifying the `without overwrite` history option when version-enabling a table keeps a persistent history of all changes made to all row versions in the table.

Valid Time Support

Some applications need to store data with an associated time range that indicates when the data is valid. That is, each record is valid only within the time range associated with the record.

If Valid Time is enabled for a table (either when or after it is version-enabled), each row contains an added column to hold the valid from and till period associated with the row as an object type. The valid time can encompass the past, present and/or the future. Before a query is performed the user sets a valid time in his session context using the procedure `SetValidTime` that acts as a filter on subsequent queries. The query only returns versions stamped with a valid time that falls within the valid time set for the session context. Workspace Manager valid time comparison operators can also be used to further refine queries within the valid time of the session context.

Merging, Refreshing and Rolling Back Workspace Changes

Rows that are inserted or changed in a workspace are only visible from that workspace until a `MergeWorkspace` or `MergeTable` command is executed. Merging a workspace involves applying changes (new row versions) made in a child workspace to its parent workspace. When a `MergeWorkspace` is executed only the current row version in the child workspace is merged into the parent workspace. That is, older historical versions in the workspace are not merged. If `MergeWorkspace` is specified with `remove_workspace` then any intermediate row versions (and history) created in the child workspace will be deleted when the child workspace is removed. The child workspace must be retained in order to retain all intermediate row versions created in the child workspace.

Rolling back a workspace deletes the changes made in the workspace to one or more version-enabled tables. User can either delete all changes made since the workspace was created or only changes made between two savepoints.

Refreshing a workspace involves applying changes made in the parent workspace to a child workspace. A workspace can either be refreshed by user request (manually) or automatically (continuously), as changes are committed or merged in the parent workspace. A continuously refreshed workspace can not have conflicts with the parent. Therefore, pessimistic locking must be used to prevent conflicts.

Conflict Detection and Resolution

When a row is changed in both the child and parent workspace (usually through the merge of another child workspace into the parent), a row conflict is created. Conflicts can be checked and resolved at any time. They are automatically detected when a merge or refresh operation is requested. The list of conflicts is presented to the user in conflict views. There is one conflict view per table. This view lists the column values of the rows in the two workspaces involved in the conflict and the common ancestor or base row.

Conflicts can be resolved row by row or for the entire table using a Workspace Manager procedure. For each conflict you can choose to keep the row from the child workspace, the row from the parent workspace, or the common base row. That is, no change, keep the original data

values for the row before the change in the parent and child occurred.. You must resolve conflicts before you can perform a merge or refresh operation.

Workspace Access Modes

You can control read and write access to a workspace by freezing and unfreezing the workspace. You can freeze a workspace in any of the following modes: no access, read-only, and one writer only. Some Workspace Manager procedures automatically freeze one or more workspaces.

Workspace Locking

Workspace Manager provides exclusive and shared version locks in addition to locks provided by regular Oracle transactions. You can enable locking on a workspace, for a user session, on specified rows, or some combination of the three. These locks are primarily intended to eliminate row conflicts between a parent workspace and a child workspace. Workspace level locking locks any row changed in the workspace. Session level locking locks any row changed by the session regardless of the workspace. Row level locking locks particular rows and can ensure all rows that must be updated are available for update.

Workspace-exclusive locks and version-exclusive locks are forms of exclusive locking that control which users can and cannot change data values, but (unlike exclusive locking) they do not prevent conflicts from occurring. Workspace-exclusive locks lock rows such that only the user that set the lock can change the values in the current workspace; however, other users in other workspaces can change the values. Version-exclusive locks lock rows such that only the user that set the lock can change the values (and that user can be in any workspace); no other users (in any workspace) can change the values.

Workspace Privileges

Workspace Manager provides a set of privileges in addition to standard Oracle database privileges. Workspace-level privileges allow the user to affect a specified workspace. System-level privileges allow the user to affect any workspace. Privileges are needed to access, create, remove, merge, rollback, compress, and freeze a workspace. The `WM_ADMIN_ROLE` role has all Workspace Manager privileges with the grant option. By default, the `DBA` role is granted the `WM_ADMIN_ROLE`. The `DBA` either grants the privileges to individual users directly, or grants the `WM_ADMIN_ROLE` role to one or more selected users, who in turn grant privileges to individual users. To grant and revoke privileges on a multiparent workspace graph, use the `GrantGraphPriv` and `RevokeGraphPriv` procedures, respectively.

Tables with nested table columns can be version-enabled if you set the new `ALLOW_NESTED_TABLE_COLUMNS` system parameter to `ON`.

Workspace Manager Events

Several types of Workspace Manager operations can be captured as events, and can be communicated to applications through the Oracle Advanced Queuing framework. Messaging features, such as asynchronous notification, persistence, propagation, access control, history, and rule-based subscription, can be used for Workspace Manager events. Support for Workspace Manager events includes the `ALLOW_CAPTURE_EVENTS` system parameter, the `SetCaptureEvent` procedure, and the `WM_EVENTS_INFO` metadata view.

Spatial Topology Support

Special techniques have been implemented for using Workspace Manager with tables in Oracle Spatial topologies. Topologies are useful when there is a high degree of feature editing and a strong requirement for data integrity across maps and map layers. A workspace can isolate a collection of changes to one or more topologies, keep a history of changes and create multiple topological scenarios for “what if” analysis in the same database.

A topology is versioned by a special form of the Workspace Manager `EnableVersioning` procedure. A topology geometry layer is added or deleted in a version-enabled topology using the `ADD_TOPO_GEOMETRY_LAYER` and `DELETE_TOPO_GEOMETRY_LAYER` procedures. These procedures have the same format and meaning as the `SDO_TOPO.ADD_TOPO_GEOMETRY` and `SDO_TOPO.DELETE_TOPO_GEOMETRY` procedures, documented in Oracle Spatial Topology and Network Data Models; however, you must use the `DBMS_WM` procedures with topology geometry layers in a version-enabled topology.

Please see the Oracle whitepaper *Oracle Database 11g Workspace Manager Support for Oracle Spatial Topology Data Model* for details.

Management Features

Workspace Manager provides a complete set of workspace semantics implemented in PL/SQL and metadata views that report on all aspects of the workspace environment. The Workspace Manager environment also can be managed from Oracle Enterprise Manager.

Removing Unwanted Versions And Workspaces

Compressing a workspace or a workspace tree deletes explicit savepoints and intermediate versions in the workspace, and minimizes the Workspace Manager metadata structures for the workspace. The compression operation reduces disk storage and improves Workspace Manager performance by reducing the number of versions involved in DML operations. It also allows users to reuse savepoint names after they are deleted. Workspaces and entire workspace trees can be removed. This deletes the workspace structure. If any unmerged versions exist in the

workspace they are deleted as well. The system parameters `commit_in_batches` and `batch_size` allow control over the compression process.

Views To Manage Workspaces

Following standard methodology for Oracle metadata views, Workspace Manager provides a number of views that give the DBA information about all aspects of the workspace environment. These views are read-only to users.

DDL Operations On Version-Enabled Tables

Data definition language commands (DDL) can be performed on version-enabled tables. DDL operations on columns, indexes, constraints and triggers are supported. To perform DDL operations on a version-enabled table, you must use Workspace Manager procedures before and after the DDL operations to ensure that Workspace Manager versioning metadata is updated.

Constraints On Version-Enabled Tables

Version-enabled tables can have referential integrity constraints, including constraints with the `CASCADE` and `RESTRICT` options. If the parent table in a referential integrity relationship is version-enabled, the child table must also be version-enabled. (The child table is the one on which the constraint is defined.) A child table in a referential integrity relationship can be version-enabled without the parent table being version-enabled. Multilevel referential integrity constraints are permitted on version-enabled tables.

Tables with unique and check constraints defined on them can be version-enabled. A `UNIQUE` constraint or unique index can be placed on a single column or multiple columns. A functional unique index can be placed on the table.

Triggers On Version-Enabled Tables

Version-enabled tables can have triggers. Per-row and whole-row triggers are supported. Triggers can only call-out to PL/SQL procedures. That is, the `action_type` must be `PL/SQL`. Per-statement, before-update and after-update triggers for specific columns are not supported. They are deactivated when versioning is enabled and are reactivated when versioning is disabled.

Import And Export Version-Enabled Tables

Database-wide and workspace-by-table level import and export operations are supported for version-enabled databases. Workspace level Import and Export is accomplished by exporting one version-enabled table at a time. The scope of the table export from the workspace is either the entire table as seen from the workspace or just the changes to the table made from the workspace.

If a database-wide operation is performed the target database must have Workspace Manager installed and must not have any version-enabled tables or workspaces (that is, other than the LIVE workspace). Other export modes (such as schema and partition) are not supported because it is not feasible to export a portion of a version hierarchy. The FROMUSER and TOUSER capabilities of the Oracle Database import utility are also not supported with version-enabled databases.

Replication Support

Multimaster replication of version-enabled tables in an asynchronous mode is supported with certain restrictions. This includes support for all workspace-related entities (such as workspaces and savepoints), operations (such as CreateWorkspace and MergeWorkspace), and DML and DDL operations on version-enabled tables.

The main restriction imposed on the replication sites is that only the master definition site in the multimaster setup can perform workspace operations and DML and DDL operations on version-enabled tables. All other sites are disallowed from performing any write operations. All read operations, such as GotoWorkspace or SELECT queries on version-enabled tables, are allowed on all sites in the replication environment.

SQL*Loader for Bulk Loading

Workspace Manager provides special procedures that enable SQL*Loader to perform bulk loading into version-enabled tables, some restrictions apply.

You can perform both direct-path and conventional-path bulk loading of data into either the latest version of any workspace or into the root version (version number 0, which is in the LIVE workspace). The root version is the ancestor of all other versions, so data in the root version is visible from all other workspaces (unless non-LIVE workspaces have updated the data).

Materialized View Support

You can create a materialized view on a version-enabled table only if you specify the complete refresh method when you create the materialized view. You cannot specify any of the following clauses in the CREATE MATERIALIZED VIEW statement: FAST (incremental refresh), ON COMMIT, or FOR UPDATE.

You cannot version-enable a materialized view or the base table of a materialized view. When the materialized view is created, its content is based on the workspace in which the session is at that time. When the materialized view is refreshed, its content is based on the workspace in which the session is when the operation is performed. When the materialized view is created or refreshed, it shows the same data in all workspaces.

Using Virtual Private Database Security

You can use Workspace Manager in conjunction with the Oracle Virtual Private Database (VPD) technology. (Virtual private databases are described in Oracle Database Security Guide.)

However, Row-level security policies are not enforced during workspace operations, such as MergeWorkspace. A call to MergeWorkspace will merge all the changes made in a workspace, not just the changes that the current user can see. Row-level security policies must be defined on a version-enabled table and all Workspace Manager views associated with the table, such as the `_LOCK`, `_CONF`, `_DIFF`, and `_HIST` VIEWS.

Support for Table Synonyms

You can specify a synonym for any Workspace Manager procedure or function input parameter that calls for a table name. When Workspace Manager looks for a table it searches and uses the first match for the specified name: a table in the specified schema, a private synonym in the specified schema or a public synonym

System Parameters for Workspace Manager

Workspace Manager provides a set of system parameters that allow a user with the `WM_ADMIN_ROLE` role to enforce global Workspace Manager-specific settings for the database. Parameters affect the use of events, multiparent workspaces, nested table columns, continuously refreshed workspaces, triggers, locking, compression, undo space, and timestamp type for history.

New Features for Oracle Database 11g

New Oracle Database feature support:

- Oracle Label Security policies are supported when tables are version-enabled, and policies can be applied/removed and enabled/disabled on version-enabled tables using the `BeginDDL` procedure.
- Partitioning operations Add, Merge and Split can be performed on version-enabled tables using the `AlterVersionedTable` procedure. All partitioning schemes are supported.
- NULL Constraints can be used to cause the foreign key of related child table rows to be set to null using the `BeginDDL` procedure.
- Index rebuilding can be done on a version-enabled table using the `AlterVersionedTable` procedure.

Performance enhancements

- User-defined hints can be specified on DBMS_WM package SQL statements to modify default optimizer hints using two new Workspace Manager procedures: AddUserDefinedHint and RemoveUserDefinedHint.
- Row level locking can be specified for Merge operations to improve concurrency by setting the new Workspace Manager system parameter: ROW_LEVEL_LOCKING. This causes a shared lock rather than an exclusive lock to be taken on the parent workspace.
- Compressing parent workspace savepoints can be deferred to a less busy time following a RemoveWorkspace operation on a child workspace by setting the new Workspace Manager system parameter: COMPRESS_PARENT_AFTER_REMOVE.
- PGA memory can be constrained for large merge operations by setting the new Workspace Manager system parameter: TARGET_PGA_MEMORY.

Ease-of-use enhancements

- Valid Time can be stored as two scalar types instead of an object type and have ValidFrom and ValidTill dates specified when a table is version-enabled by setting new Workspace Manager system parameter USE_SCALAR_TYPES_FOR_VALIDTIME and using the EnableVersioning procedure, respectively. Use the AlterVersionedTable procedure to change the way Valid Time is stored for a table that is already version-enabled.
- History and valid time rows can have a unique key and the workspace which creates/retires a history row track can be persistently archived by setting the new Workspace Manager system parameters: ADD_UNIQUE_COLUMN_TO_HISTORY_VIEW & KEEP_REMOVED_WORKSPACES_INFO, respectively.
- Removed workspace information can be found in two new Workspace Manager views: ALL_REMOVED_WORKSPACES and USER_REMOVED_WORKSPACES.
- A set of version-enabled tables involved in a Referential Integrity Constraint can be merged at the same time by providing a list with the MergeTable procedure.
- Workspaces have a unique ID number, which can be found in the ALL_WORKSPACES and USER_WORKSPACES views.

Conclusion

Workspace Manager, a feature of Oracle Database allows application developers and DBAs to manage current, proposed and historical values for data in the same database. It can isolate a collection of changes to production data, keep a history of changes to data and create multiple data scenarios for “what if” analysis. It can save money, time and effort over traditional approaches. The Oracle Database 11g release of Workspace Manager includes additional database feature support, and performance and ease of use enhancements.



Oracle Database 11g Workspace Manager
Overview
September 2009
Author: Bill Beauregard
Contributing Authors: : Chuck Murray and Ben
Speckhard

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com



| Oracle is committed to developing practices and products that help protect the environment

Copyright © 2009, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.