Oracle Database 10g Workspace Manager Overview

An Oracle White Paper
May 2005

Oracle Database 10g Workspace Manager Overview

Oracle Database 10g Workspace Manager Overview

EXECUTIVE OVERVIEW

Applications and DBA operations often work with more than one version of the data. Three common reasons to have multiple data versions are concurrency, auditing and scenario creation. Concurrency allows one or more users to make changes to the same data in isolation while others are using the current version of this data. Auditing allows earlier versions of the data to be kept for weeks, months or years and allows users to go back in time to see how the database looked in the context of the changes. Scenario creation allows a base data set to be altered for "what if" analysis.

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 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 labor over traditional approaches.

INTRODUCTION

Workspace Manager is a feature of the Oracle Database for application developers and DBAs. It manages current, proposed and historical values for data in the same database using workspaces. A workspace is a virtual environment. It logically groups collections of changes (that is new row versions) that are physically contained in one or more version-enabled tables. Users must set their session context to a particular workspace to access new versions created from that workspace. All other users access the current production versions of the data until the new versions are explicitly merged with production data or discarded, thus providing maximum concurrency.

Workspace Manager saves money by reducing the amount of hardware and corresponding software licensing needed to manage multiple versions of the data, as compared to scenarios where data is copied and synchronized. It saves time by allowing concurrent access to current, proposed and historical data. It saves labor by allowing a single point of update and management for all 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.

Potential uses for Workspace Manager include isolating a collection of changes to production data, keeping a history of changes to data and creating multiple data scenarios for “what if” analysis. The following three case studies illustrate some of the ways in which customers use Workspace Manager today.

Creating An Operational Data Store

A major HMO is building an operational data store (ODS) to support 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.

Oracle Database with Workspace Manager was chosen because it only versions changed rows, requiring less storage than a table or database copy. And, 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 if needed.

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 Locator 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 Time Based "What if" Analysis

A major railway company is developing a strategic planning application with Workspace Manager. It is intended to manage projects related to the construction, upgrade and repair of its assets. These assets include control centers, stations, railway lines and bridges. 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 you 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-enabled 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. Creation of new row versions is done by conventional Oracle short transactions, ensuring integrity of versioned data. The versioning infrastructure is not visible to the users of the database, and application SQL data manipulation statements (DML) to select, insert, modify, and delete data continue to work in the usual way with version-enabled tables.

Workspace Manager implements these capabilities by renaming the table, 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 that one or more users can share to make changes to the data in the database. Users set their session context to a workspace before making changes to data. When a change is made to a row from a workspace a new version of the row is created. Subsequent changes to the row are applied to the same version until a savepoint is created (see Savepoints below).

A workspace logically groups collections of changes, (that is, new row versions) which are physically contained in one or more version-enabled tables. Users must set their session context to a particular workspace to access new versions created from that workspace. All other users access the current production versions of the data until the new versions are explicitly merged with production data or discarded, thus providing maximum concurrency.

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.

There can be a hierarchy of workspaces in the database. By default, when a workspace is created, it is created from the topmost database workspace, which is always called "LIVE". A child workspace can 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 see all of the changes made in or make a change that needs to be visible in, two or more workspaces.

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

If a child workspace has two or more parent workspaces it becomes a multi-parent workspace. A multi-parent workspace can be used to see all of the changes made in two or more parent workspaces without needing to merge the parent workspaces. It can also be used to merge changes from the child into two or more parent workspaces.

Savepoints

Savepoints 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 are usually created in response to a milestone like a business event, such as the completion of a design or the end of a billing 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.

Keeping a 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 works with savepoints to provide 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 or over-write values in the row version with the most recent changes and timestamp. If you specify the without overwrite history option when version-enabling a table, you keep 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 the validity of the data. 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. 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

Workspaces can be merged, refreshed or rolled back. Merging a workspace involves applying changes (new row versions) made in a child workspace to its parent workspace(s). Rolling back a workspace involves deleting changes made to a single table or to all version-enabled tables in the workspace. User can either delete all changes made since the workspace was created or only changes made after a savepoint.

Refreshing a workspace involves applying changes made in the parent workspace(s) 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.

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 data 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.

Conflicts are resolved manually 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). 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. If a workspace is frozen, the ability of users to access the workspace and to make changes to rows in version-enabled tables from the workspace is restricted. 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

In addition to locks provided by regular Oracle transactions, Workspace Manager provides exclusive and shared version locks. These locks are primarily intended to eliminate row conflicts between a parent workspace and a child workspace. You can enable locking on a workspace, for a user session, on specified rows, or some combination of the three.

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 multiparent graph workspaces, use the `GrantGraphPriv` and `RevokeGraphPriv` procedures, respectively.

Tables with nested table columns can now 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 Queueing 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.

The `ADD_TOPO_GEOMETRY_LAYER` and `DELETE_TOPO_GEOMETRY_LAYER` add and delete a topology geometry layer in a version-enabled topology. 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.

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 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 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.

Bulk Loading Into Version-Enabled Tables

You can use SQL*Loader to perform bulk loading into version-enabled tables, but you must also use some new Workspace Manager procedures, and 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).

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.

CONCLUSION

Applications and DBA operations often need to work with multiple versions of data for reasons of concurrency, auditing and scenario creation. Workspace Manager is a feature of the Oracle Database. It 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 labor over traditional approaches.

Oracle Database 10g Workspace Manager Overview
May 2005
Author: Bill Beauregard
Contributing Author: Chuck Murray

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