ORACLE DATABASE 12c

An Oracle White Paper

June 2013

Oracle Database 12*c*: Workspace Manager

ORACLE

Introduction 2
Workspace Manager 2
Use Cases..... 3
Application Development Features 4
Management Features 9
Enterprise Features Supported in Oracle Database..... 10
Oracle Exadata Database Machine 12
Support for Oracle Multitenant..... 13
Workspace Manager Partners 13
Conclusion 13
Appendix: New Workspace Features for Oracle Database 12*c* 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 of copying data.

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 are 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 resources and time needed to manage multiple versions of the data 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 session context to different workspaces. 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 does not require changes to application SQL statements to access version-enabled tables.

The next section introduces case studies that illustrate some of the ways in which customers use Workspace Manager.

Use Cases

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 with Oracle Spatial and Graph to manage current, proposed and historical values of location 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` procedure enables users use to see the database as it existed at any point in time.

Creating an Operational Data Store

An operational data store (ODS) for a major healthcare system supports key operational business processes. It aggregates transaction-processing data from multiple legacy applications and provides subject-oriented, integrated, near real time, 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. 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 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` procedure 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.

Performing "What if" Analysis

A major transportation company developed a strategic planning application with Workspace Manager. It manages 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 set of PL/SQL procedures 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. Tables with nested table columns can be version-enabled by setting a system parameter, `ALLOW_NESTED_TABLE_COLUMNS`.

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 to data that are new row versions 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. Workspaces have a unique ID number that can be found in the `ALL_WORKSPACES` and `USER_WORKSPACES` views.

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 the Savepoints section 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 cannot 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 the `GotoDate` procedure sets session context to a historical time. The `GotoSavepoint` procedure sets session context to a particular savepoint. A subsequent `SELECT` 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 that is only visible in the current workspace. 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 using the `without_overwrite` option or over-write values in the row version with the most recent changes and the current timestamp using the `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.

History and valid time rows can have a unique key and the workspace which creates and retires a history row track can be persistently archived by setting system parameters, `ADD_UNIQUE_COLUMN_TO_HISTORY_VIEW` and `KEEP_REMOVED_WORKSPACES_INFO`.

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 valid 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.

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 a system parameter, `USE_SCALAR_TYPES_FOR_VALIDTIME`. Use the `AlterVersionedTable` procedure to change the way valid time is stored for a table that is already version-enabled.

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` procedure is executed. Merging a workspace involves applying changes, the new row versions made in a child workspace to its parent workspace. When a `MergeWorkspace` procedure 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 the `remove_workspace` parameter is used with the `MergeWorkspace` procedure then any intermediate row versions and their history 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. 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.

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 manually by user request or automatically and continuously as changes are committed or merged in the parent workspace.

Conflict Detection and Resolution

When the same row is changed in 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. Conflicts are automatically detected when a merge or refresh procedure 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 one row at a time or on 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 which means keeping 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, one writer only, and wm-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, a user session, a set of 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. Row level locking can be specified for Merge operations to improve concurrency by setting a system parameter, `ROW_LEVEL_LOCKING`. This causes a shared lock rather than an exclusive lock to be taken on the parent workspace.

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 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. Privileges can also be granted on a multiparent workspace graph.

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. Support for Workspace Manager events includes the `ALLOW_CAPTURE_EVENTS` system parameter, the `SetCaptureEvent` procedure, and the `WM_EVENTS_INFO` metadata view.

Performance

User-defined hints can be specified on `DBMS_WM` package SQL statements to modify default optimizer hints using two procedures, `AddUserDefinedHint` and `RemoveUserDefinedHint`.

PGA memory can be constrained for large merge operations by setting a Workspace Manager system parameter, `TARGET_PGA_MEMORY`.

Spatial Topology Support

Oracle Spatial and Graph topologies can be version-enabled. 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 Workspace Manager `ADD_TOPO_GEOMETRY_LAYER` and `DELETE_TOPO_GEOMETRY_LAYER` procedures. These procedures have the same format and meaning as the procedures of the same name, documented for Oracle Spatial and Graph 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 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` control the compression process.

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.

Views to Manage Workspaces

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

Enterprise Features Supported in Oracle Database

Oracle Database provides powerful, reliable support for an organization's mission-critical applications. These enterprise features enrich Workspace Manager capabilities.

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. Index rebuilding can be done on a version-enabled table using the `AlterVersionedTable` procedure.

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 being the one on which the constraint is defined. A child table in a referential integrity relationship can be version-enabled when the parent table is not. 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

Workspace Manager supports the import and export of version-enabled tables in any of the following ways: a full database import and export, an import and export that only includes the schemas required by Workspace Manager, or a workspace-level import and export through Workspace Manager procedures.

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 other than the `LIVE` workspace. Other export modes 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.

A limited export and import can be performed that includes all schemas related to version-enabled tables and workspaces, as well as any Workspace Manager metadata, but excludes all other schemas. The restrictions on a database-wide operation apply.

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.

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.

Partitioning

Partitioning operations Add, Merge and Split can be performed on version-enabled tables using the `AlterVersionedTable` procedure. All partitioning schemes are supported.

Advanced Security

Workspace Manager supports Oracle Label Security policies when tables are version-enable, and policies can be applied, removed, and enabled and disabled on version-enabled tables using the `BeginDDL` procedure.

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.

Oracle Exadata Database Machine

Engineered systems provide high performance, high bandwidth, and massive parallelism with enormous capacity to address the challenges faced by high volume workloads. Combining Oracle Database Workspace Manager with Oracle Exadata Database Machine performance and scalability delivers an ideal platform for versioning in the most demanding applications.

Support for Oracle Multitenant

Oracle Multitenant is an option for Oracle Database that enables database consolidation without changes to applications. Designed for the Cloud, it allows many databases to be managed as one, while retaining the isolation and resource prioritization of separate databases. The multitenant architecture consolidates multiple Oracle Databases (each referred to as a pluggable database) to run under a single occurrence of Oracle Database software (referred to as a multitenant container database). Architectural separation is enforced between each pluggable database (user data and metadata) and its multitenant container database (Oracle metadata). Pluggable databases are compatible with traditional Oracle Databases not in a multitenant container database.

Oracle Database Workspace Manager functions transparently in a multitenant architecture; Workspace Manager applications benefit from the efficient administration of one multitenant container database, and the separation and resource prioritization allowed by multiple pluggable databases.

Workspace Manager Partners

Oracle's longstanding commitment to depth and breadth of partnerships provides users flexibility and the widest possible choices. Developers and IT managers can select best of breed tools and applications to meet their industry and organization-specific requirements and rapidly deploy scalable, secure enterprise solutions. Workspace Manager is supported by most GIS vendors. A list of partners is available on the Oracle Technology Network.

Conclusion

Workspace Manager, a feature of Oracle Database provides versioning capabilities for Oracle Database. It addresses the needs of application developers and DBAs to manage current, proposed and historical values for data in the same database. It is used by hundreds of customers to isolate a collection of changes to production data, keep a history of changes to data or create multiple data scenarios for "what if" analysis. It can save money, time and effort over traditional approaches to copying data.

Appendix: New Workspace Features for Oracle Database 12*c*

- Perform more focused import and export of only the schemas required by Workspace Manager with new procedures, `Export_Schemas` and `Import_Schemas`.
- Create a duplicate of a version-enabled table in a non-versioned form in another schema or in another database that can be version-enabled or used in its non-versioned form with the new procedure, `GetOriginalDDL`.

- Bulk load data that is visible in all workspaces by loading the data into the first version (version number 0) of version-enabled table, which is in the `LIVE` workspace.
- Bulk load more easily, the `CommitBulkLoading` automatically uses the same settings as the `BeginBulkLoading` procedure and the `GetBulkLoadVersion` procedure is no longer needed.
- Control the amount of undo space used during the `DisableVersioning` procedure with a new parameter, `undo_space`.
- Remove both versions in a `ResolveConflicts` procedure when there is no common base version (insert-insert conflicts) with a new parameter, `resolve_base_ne`.
- Perform more DDL on version-enabled tables: add and remove supplemental logging on a version-enabled table; rename a column of a version-enabled table; and modify the compression options on a version-enabled table.
- Manage Workspace Manager operations with new data dictionary views and changes to existing views.
- Sort rows in a SQL query on a version-enabled table based on their valid time and return distinct results using the `wm_period_map` member function for columns of type `WM_PERIOD`.
- Security is improved for procedures running Workspace Manager procedures and for queries on Workspace Manager views; the privilege checks now take into account whether the procedure has definer's rights or the rights of the database user whose privileges are currently active.

Oracle Database 12*c*: Workspace Manager

June 2013
Author: Oracle
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

0809