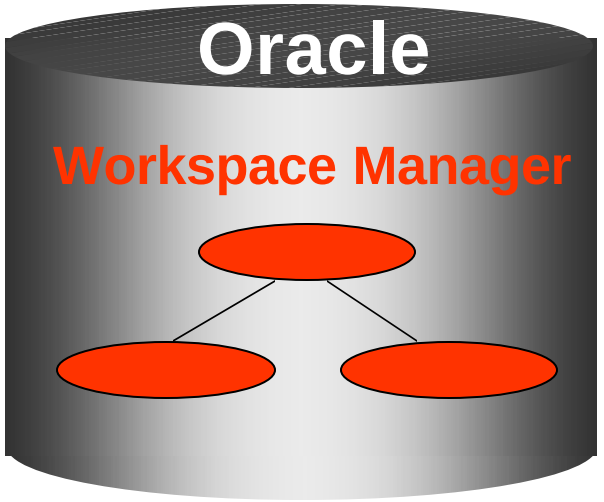
Oracle9*i* Workspace Manager Technical Overview



Agenda

- Long Transactions
- Oracle Workspace Manager
- Application Scenarios
- Mechanics, Features and Benefits
- Setup and Management
- Application Development

What is a Long Transaction?

- A group of atomic data operations comprising a complete operational task
- Occurs over an extended period of time
- Does not require the data be locked over the entire period of time.

Capabilities of a Long Transaction System

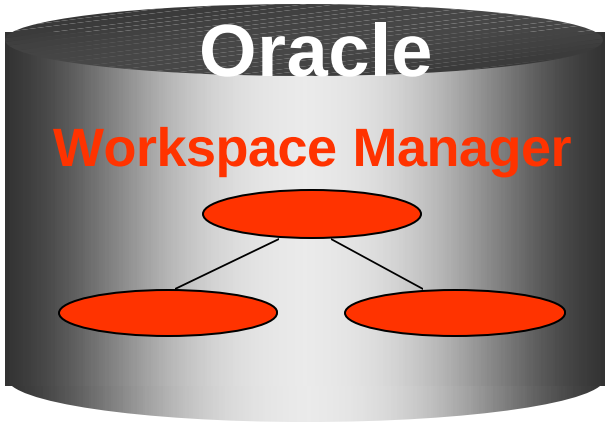
- Permits multiple versions of the data to be independently developed, accessed, compared and kept or discarded
- Enables multiple users to simultaneously edit and query the same data with deferred conflict detection
- Isolates a collection of updates made over time until merged into production
- Allows updates to be seen in context of entire database
- Detects and resolves conflict between versions
- Implemented as a series of short transactions

How do Long and Short Transactions Differ?

Long Transaction	Short Transaction
Completes over days or weeks	Completes in seconds or less
Likelihood of conflict is low - Optimistic concurrency permits conflicts (resolved before merge)	Likelihood of conflict is high - Pessimistic locking prevents conflicts
Selective versioning of tables • update creates a new row version • Each update is part of a short transaction	Data is in a single state
Collections of updates isolated in workspaces until merged into production	Updates accessible upon commit
Multi-user updates	Single user updates

Oracle Workspace Manager

- Long Transaction system for Oracle9*i*
  - Isolate a collection of changes to production data
  - Model data scenarios for “what if” analysis
  - Create multiple editions of data
  - Keep a history of changes to data
- Enables project collaboration
- No changes to application SQL (DML)
- Can also be used with Oracle8i

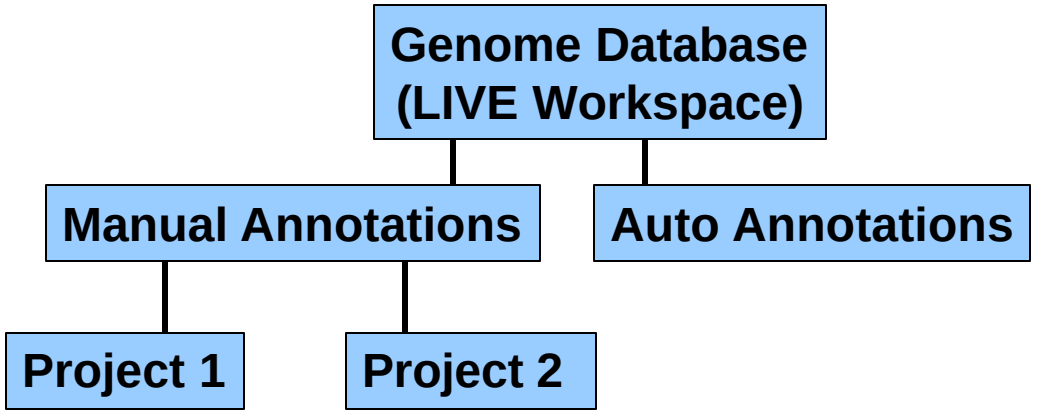


Example: Grouping a Collection of Updates

Manage a collection of changes as a unit before incorporating them into production data

- Review and rollback unapproved changes
- Isolate changes from other users
- Give other users continued access to all production data
- Organize changes in workspace hierarchies
- Explicitly check-out rows for update to assure availability

Example:
Life Sciences - Discovery and quality processes



Other Examples of Grouping a Collection of Updates

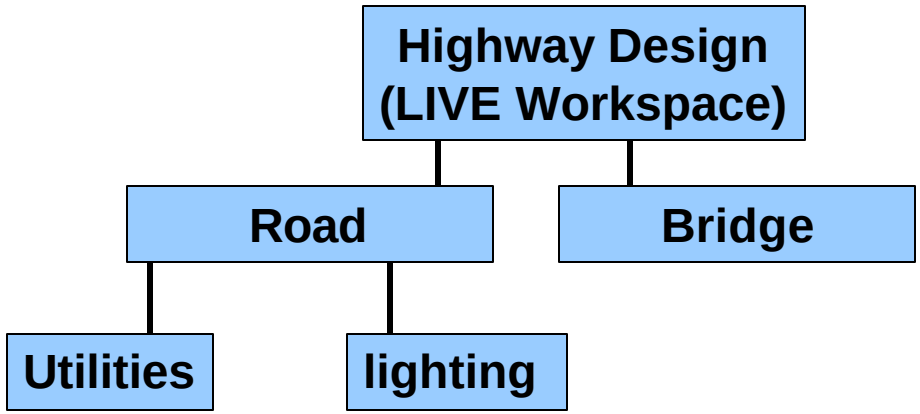
- Import legacy data with validation and testing
- Batch changes to personnel data with review and approval
- Annotate seismic data for oil and gas exploration
- Version a collection of rich media to find optimal combination of resolution, cropping and touch-ups before publication

Example: Collaborative Development

Manage content for a collaborative project

- Allow teams to share a collection of content and changes
- Enable concurrent development on multiple projects
- Control access to workspaces and workspace operations
- Prevent, or detect and resolve data conflicts
- Control ability to read and write in a workspace

Example:
Engineering -
Design projects

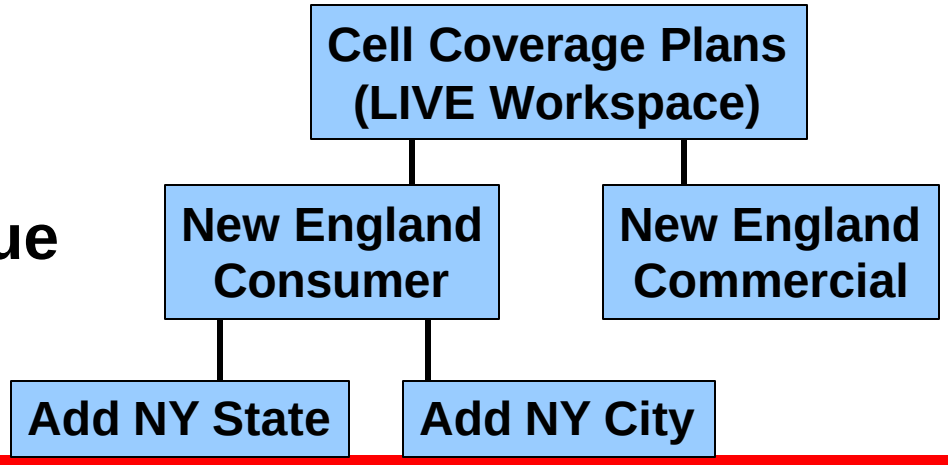


Example: Versioning a Data Set

Version a common data set for "What if" analysis or to create multiple editions of the data for publication

- Organize changes in hierarchical workspaces
- View changes in context of unchanged data
- No duplication of unchanged data
- Conflict detection and resolution for simultaneous changes
- Explicit row check-out for update to assure availability

Example:
Telecommunications -
Optimize cell plan revenue

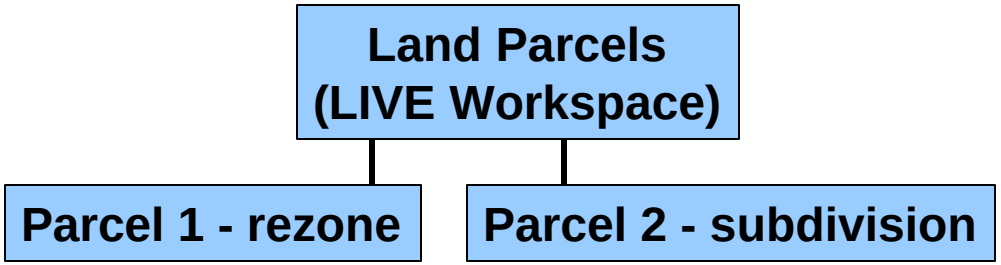


Example: Persistent History of Changes

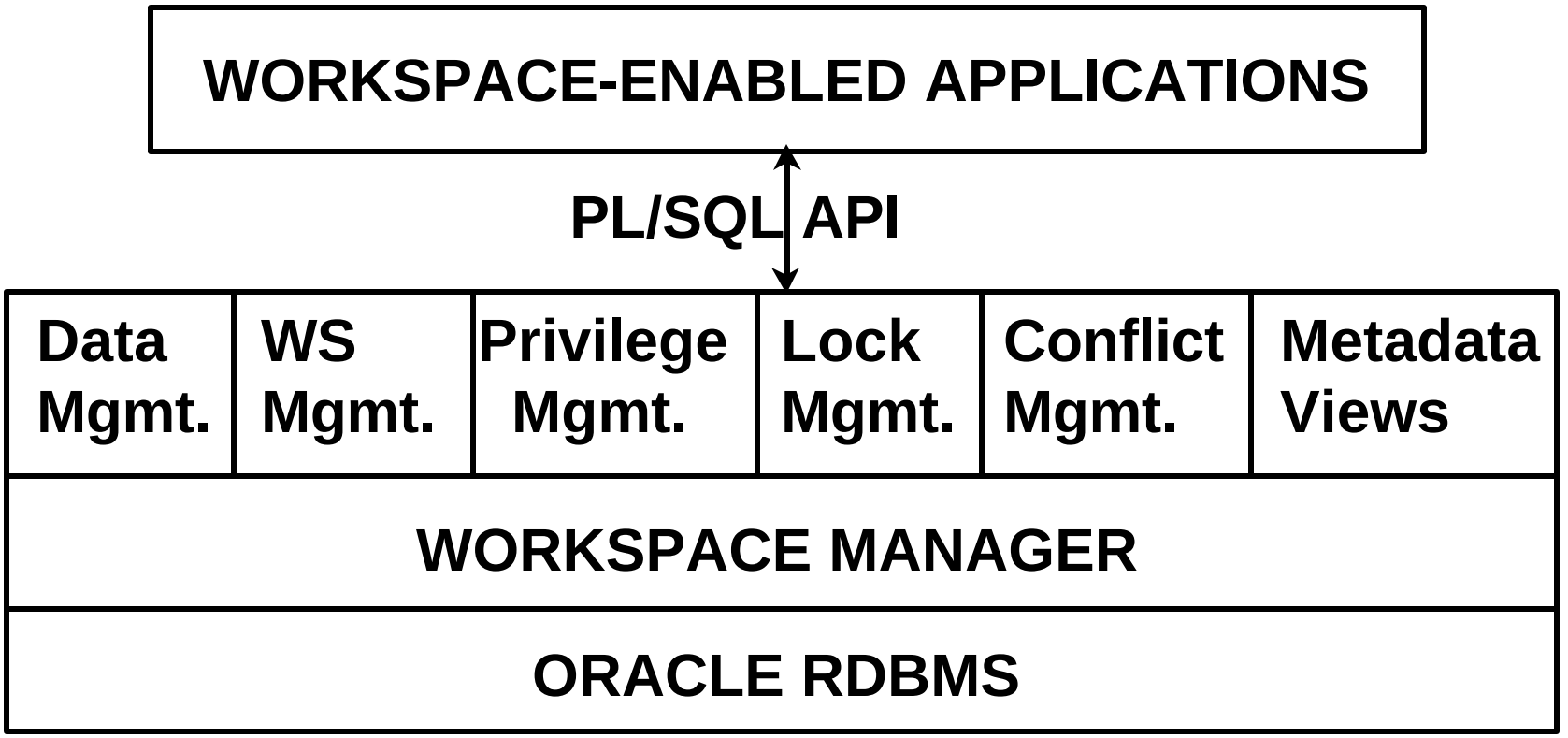
Keep a persistent history of changes to data

- Use savepoints to identify change milestones
- Keep history of all or latest changes to a row version
- Navigate versions to any savepoint or date
- Rollback unapproved changes to a row or table in a workspace

Example:
Land Information Mgt. - Support regulatory requirements for history



Workspace Manager Architecture



Workspace Manager Mechanics

- **Version-enabling a table** allows rows to be versioned
- **Implicit and in place versioning** of row versions
- **Workspace** logically groups a collection of row versions
- **Transactionally consistent view of changes** in the context of the entire database
- **"`LIVE`" Workspace** contains production data
- **Savepoint** groups collection of row changes in a workspace
- **Savepoint** causes subsequent changes to row to be implicitly captured as a new version
- **Child workspace** causes an implicit savepoint in the parent
- **History option** creates a copy of a version for each change

Workspace Manager Operations

Add PL/SQL APIs to the application or perform workspace operations from Enterprise Manager

- **Workspace:** create, refresh, merge, rollback, remove, goto, compress, alter
- **Savepoints:** create, alter, goto*
- **History:** goto date*
- **Privileges:** access, create, delete, rollback, merge
- **Access Modes:** read, write, management, none
- **Locks:** exclusive and shared
- **Differences:** compares savepoints and workspaces
- **Detect / Resolve Conflicts:** choose version to merge

* Application API only

Benefits for Developers and DBAs

- Rich concurrency and security model
- Complete set of workspace semantics
- Requires no changes to application SQL DML
- Tightly integrated with Oracle DBMS
- Efficient storage of versioned data
- Easy to manage via Oracle Enterprise Manager

Oracle Workspace Manager Setup and Management

Workspace Manager Setup and Management

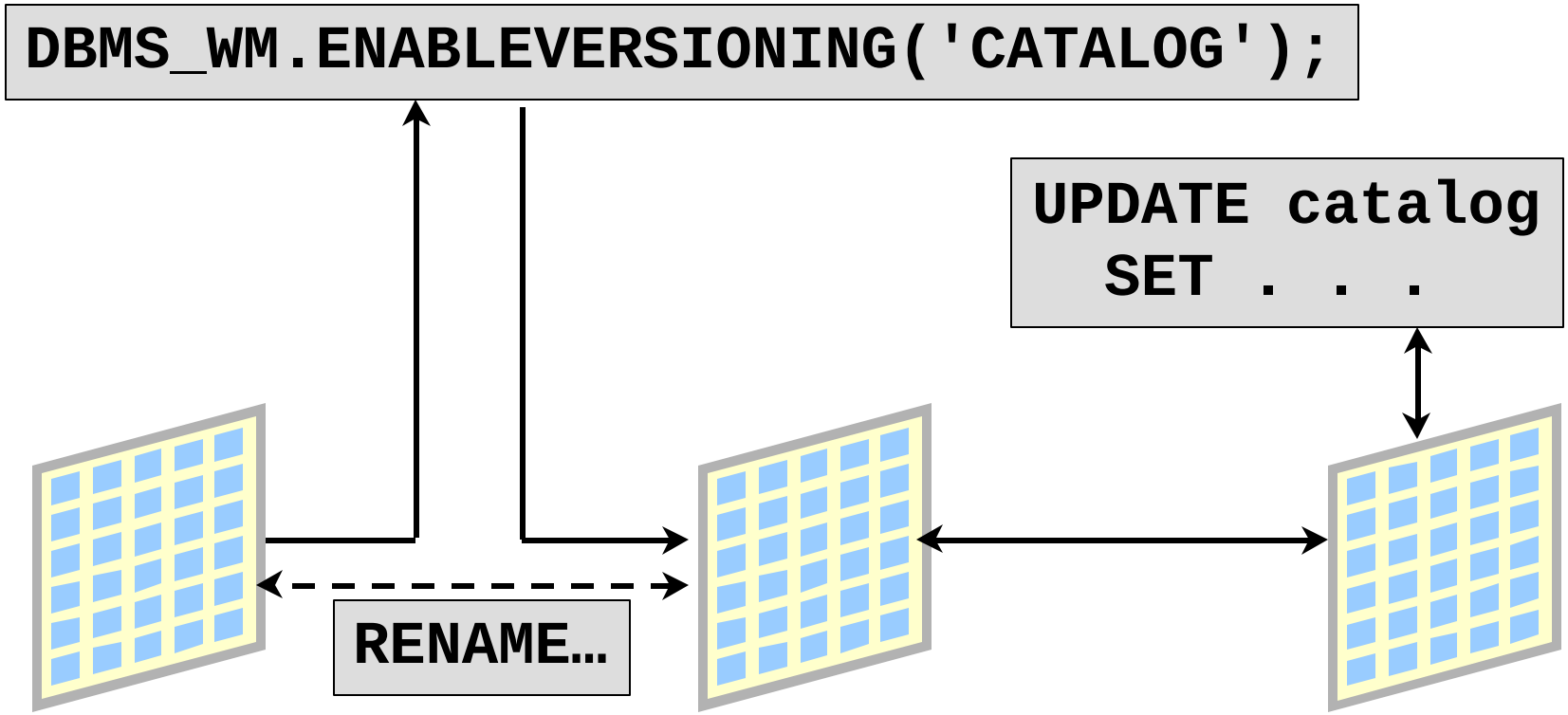
Overview:

- Assign the workspace role **`WM_ADMIN_ROLE`**
- Version-enable one or more tables
- Grant workspace privileges to users
- Use metadata views to manage workspaces
- Compress workspaces to reduce storage and improve performance

Workspace Manager Administrator Role

- Oracle installer automatically installs Workspace Manager and creates the **WM_ADMIN_ROLE**
- The **WM_ADMIN_ROLE** role has all Workspace Manager privileges
- By default, the DBA role is granted **WM_ADMIN_ROLE**.
- The DBA can grant privileges or can grant the **WM_ADMIN_ROLE** role to one or more users to grant the privileges.

Version-Enable a Table



CATALOG:
base table

CATALOG_LT:
Renamed base table with four new columns

CATALOG:
view with instead-of triggers

Guidelines for Version-Enabling Tables

- Version-enabled table must have a primary key
- A table can be version-enabled by the table owner or by a user with **WM_ADMIN_ROLE**
- Tables owned by **SYS** cannot be version-enabled
- The history option
  - allows the end-user to record all, or just the latest changes made to a version-enabled table
  - Allows user to goto date to see changes as of a point in time
  - creates a time-stamped copy of a version for each change or just the latest change

Disable Versioning for a Table

Performed by the table owner or by a user with the `WM_ADMIN_ROLE`

- Disable versioning when changes to the version-enabled table are completed.
- Improves performance
- Workspace hierarchy and savepoints remain
- The latest version of each row in **`LIVE`** workspace remains

Grant Workspace Privileges to User

- **Workspace Privileges: `ACCESS`, `CREATE`, `REMOVE`, `MERGE`, and `ROLLBACK`**
- Privileges in the form of **`priv_WORKSPACE`** allow the user to affect a specified workspace
- Privileges in the form: **`priv_ANY_WORKSPACE`** allow the user to affect any workspace

Workspace Views Used to Manage All Elements of the Workspace

Metadata read-only views describe:

- Version-enabled tables:

  Conflicts, Differences, Locks, History & Multiworkspace
- Workspaces
- Savepoints
- Users
- Privileges
- Locks
- Conflicts

Compress Workspace or Workspace Tree

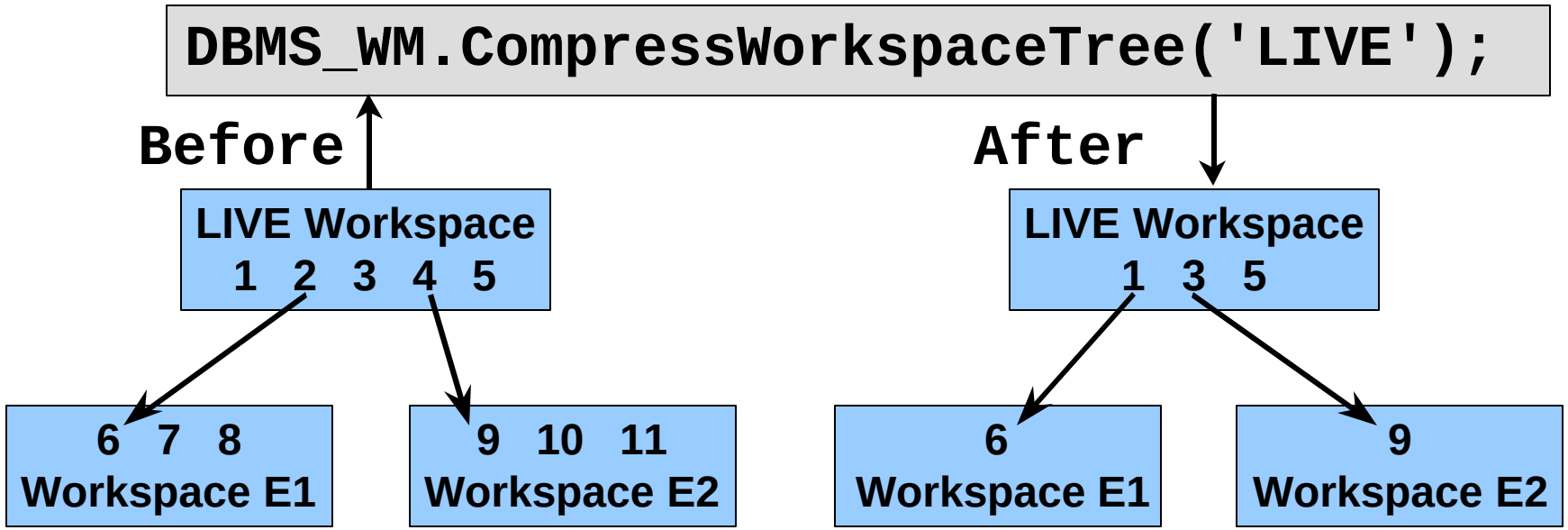
- Collapses workspace tree into optimum configuration
- Compress when the explicit savepoints in the affected workspaces are no longer needed

Compress Workspaces to...

- Reuse Savepoint names
- Improve Workspace Manager performance
- Reduce the amount of disk space used

Compress Workspace Tree Example

- Compress **LIVE** and descendents
- Keep versions 8 and 11 and renumber them as versions 6 and 9



Using Workspaces in an Application

Using Workspaces in an Application

Overview:

- Create one or more workspaces and goto workspace
- Determine locking for workspace, session, & specific rows
- Update and Insert data in a workspace
- Set savepoints to group and operate on sets of changes
- Using history to Goto date
- Change a workspace's access mode
- Rollback unapproved changes
- Refresh workspace w/ changes in another workspace
- Detect and resolve conflicts between workspaces
- Merge workspaces

Create a Workspace

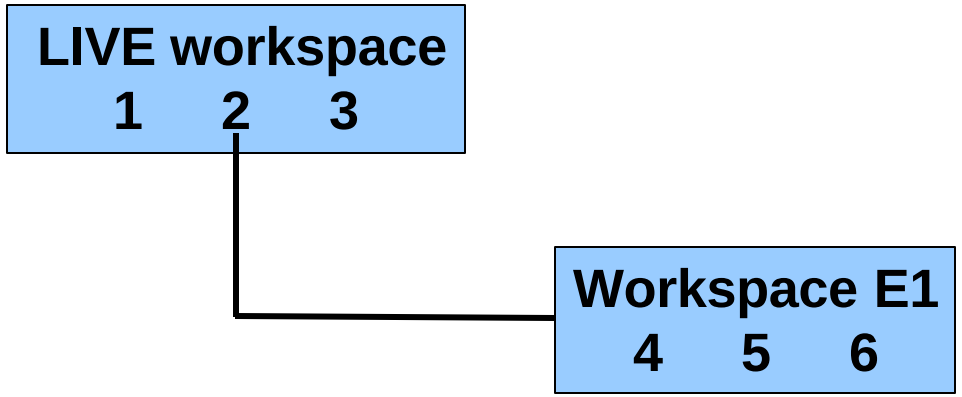
A workspace logically groups and isolates a collection of row versions representing changes to data in one or more version-enabled tables**.**

Create a workspace when...

- Creating something new with a new group of users
- Freezing a set of changes
- Business process groups/isolates a set of changes
- Creating multiple scenarios for what if analysis
- Managing updates for multiple separate collaborative projects
- Enabling two or more people to work on same row

Create a Workspace (continued)

- A new workspace is a child of the current workspace
- Creates an implicit savepoint in the current workspace. This ensures workspace users have a transactionally consistent view of all data as it existed when the workspace was created.
- Example: Implicit savepoint 2 is created when workspace E1 is created.



Workspace Locking

Eliminates row update conflicts between two workspaces

- Locks rows versioned in a workspace so they can't be changed outside the workspace
  - **Exclusive** lock prevents changes by any other user
  - **Shared** locks allow other users in the workspace to change the row
- Used with conventional Oracle short transactions locks

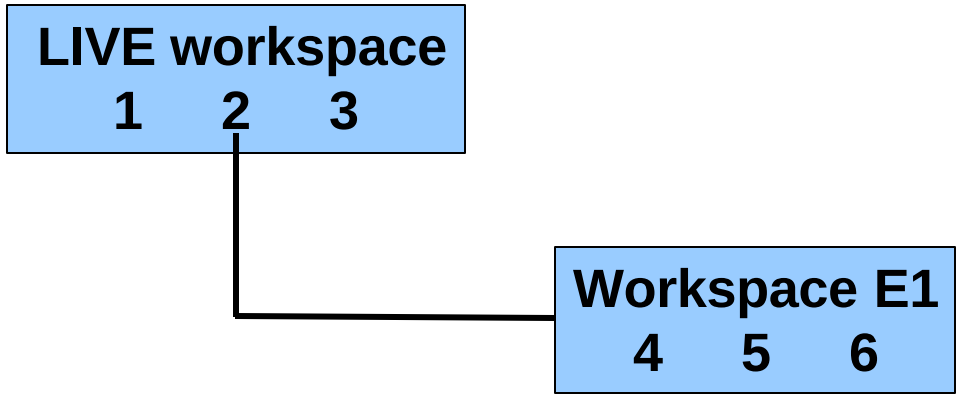
Session and Row Locking

- **Session Locking**
  - Use when user works in multiple workspaces
  - Locks all rows in all workspaces versioned by a session
- **Row Locking**

  Use to:
  - proactively locks rows for update
  - Lock updated rows to prevent additional updates

Associate a User with a Workspace

- At login, the user is placed in the **LIVE** workspace
- **GOTOWORKSPACE** procedure moves the current user session to the destination workspace
- No changes to application SQL needed to create versions
- Example: User is set in Workspace **E1.** All subsequent modifications to data by the user take place on the latest row versions in the E1 workspace.



Workspace Savepoints

A Savepoint groups a set of changes in a workspace and acts as a firewall to which changes can be rolled back..

- **Savepoint operations**: create, alter, goto, compare, rollback to and delete
  - **Goto a savepoint** to see the state of the data at that point
  - **Compare savepoints** to see differences between versions
  - **Rollback to a savepoint** to remove unapproved data changes

When to Create a Savepoint

Savepoints and workspace accomplish similar objectives. Both allow users to:

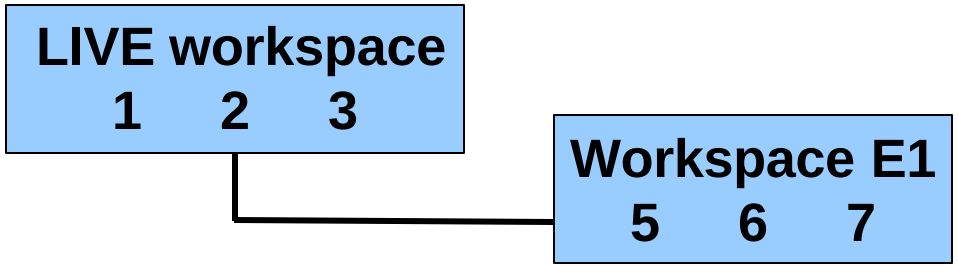
- Group a set of changes
- Compare changes in different row versions
- Rollback a set of changes

Create a savepoint (instead of a workspace) when...

- a milestone is reached in the project workspace
- making another set of changes in the same workspace
- workspace users need immediately access to changes. Changes in another workspace aren't visible in the current workspace until merged or refreshed (more on this later).
- archiving a set of changes (creating a sync point)

Examples of Implicit and Explicit Savepoints

- **Two types of savepoints**
  - –**Implicit**: created automatically in the parent when a child workspace is created
  - –**Explicit**: created by a user to group a set of changes
- **Implicit savepoint** and Version 2 is created automatically by Workspace Manager when Workspace **E1** is created
- **Explicit savepoint** is created on version 6 by a user. This allows changes in verson 7 to be rolled back to version 6



Compare Savepoints: Find Differences

Find differences in values in version-enabled tables for two savepoints in the same workspace or two different workspaces and their common ancestor or base row.

- Status values can be either either: no change, updated, deleted, inserted or nonexistent.
- **Example**: in savepoint **`e1_focus_1`**, the value for **`BUDGET`** changed to 1.5 from the base row value of 2.

```
MKT_ID BUDGET WM_DIFFVER                WM_CODE
------ ------ -------------------- ------
1       2     DiffBase               NC
1       1.5   e1_focus_1, LATEST U
1       2     e1_focus_2, LATEST NC
```

Delete a Savepoint

- Delete a savepoint when you no longer need to….
  - uniquely identify a set of changes in a workspace
  - go to it to see the state of the data at that point
  - rollback unapproved changes made since the savepoint
- Deleting a savepoint enables you to:
  - Reuse a savepoint name after it is deleted
  - Improve performance of Workspace Manager operations
  - Decrease disk storage used for Workspace Manager structures

Use the History Option

The History option and Savepoints both allow users to see the state of the data as it existed at a particular point.

Both can be used together in the same application

Use the History option to...

- create a persistent, time-stamped record of all, or just the latest, change made to a row in a version-enabled table
- goto date to see changes as of a point in time (rather than as of a savepoint milestone)

Freeze a Workspace

Freezing a workspace specifies the kind of user access allowed to the workspace:

- **NO_ACCESS** is the default
- **READ_ONLY** allows all workspace users to read
- **1WRITER** enables a single writer, all readers
- **WM_ONLY** is implicitly set by workspace operations to allow only workspace operations

Discard Unapproved Changes

Rollback operation discards unapproved changes...

- made to a table within a workspace
- made after a specified savepoint within a workspace
- all changes in the workspace

- A workspace rollback does not remove the workspace
- A savepoint rollback does require removal of all intervening implicit savepoints in the workspace.
  - Remove implicit savepoints by merging child workspaces.

Share Changes between Workspaces

The Refresh Workspace operation allows users in the current child workspace to see changes made in the parent, including those merged from other children of the parent, since the child workspace was created or last refreshed

Two types of Refresh...

- **Periodic (Manual) Refresh** (default)
  - User requested, it refreshes all changes from parent
  - Conflicts must be resolved before refresh occurs

Use Periodic Refresh when…

- Child workspace should not automatically see changes in the parent workspace
- Child workspace only refreshes after an event

Share Changes (continued)

Continuous (automatic) Refresh is an option that allows a workspace to immediately see changes made in the parent.

- Specified when the workspace is created
- Requires workspace locking to prevent conflicts (conflicts must be resolved before a refresh)
- Workspace must be a child of **`LIVE`** and have no children

Use Continuous refresh when..

  - **`LIVE`** is frozen and changes are only made in child workspaces
  - Child workspaces do not version the same rows
  - Users in a workspace want to see changes made in other related workspaces immediately

Detect and Resolve Workspace Conflicts

- Conflict exists when the same row is changed in both the parent and a child workspace
- Conflicts are automatically detected by Workspace Manager when a workspace attempts to merge with, or refresh from the parent workspace
- Merge row or table in a workspace, or entire workspace
- Conflicts must be resolved before merge and refresh operations can be successful
- Workspace Manager permits conflict resolution by displaying the contents of the: **`BASE`** (common ancestor)**`,`** and the **`CHILD`** and **`PARENT`** row versions, and allowing user to choose which to keep.

Steps to Detect and Resolve Conflicts

- Request detection of conflicts - this creates a conflict view for every version-enabled table
- Query to show the conflicts for that table
- Start a conflict resolution session
- Resolve conflicts by looping through all conflicts
- Commit or rollback conflict resolution choices

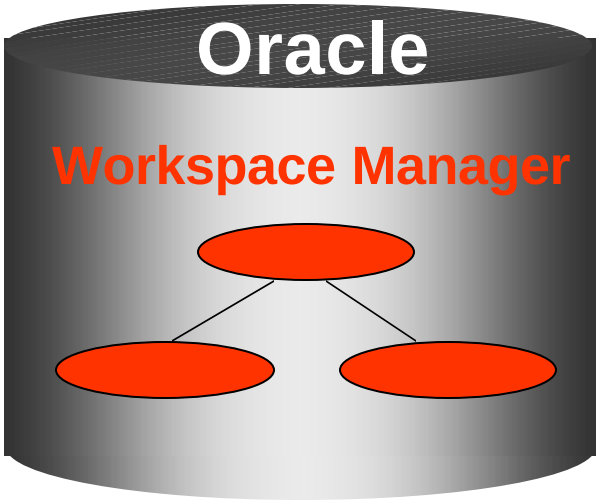
Make Workspace Changes Public

- Merge workspace operation applies changes in a child workspace to its parent workspace.
- Merge changes made to a table or specific rows in a workspace, or all changes in the workspace
- Conflicts must be resolved first.
- Optionally remove the workspace.

Oracle Workspace Manager Summary

Long Transactions: Version Rows and group collections of row versions in Oracle9*i* workspaces

- **Implicit & in place versioning -** changes cause new row versions in same table
- **No changes to application SQL (DML)**
- **Transactionally consistent**
- **Isolates content for collaborative projects**
- **Supports Oracle8*i***



Add workspaces to new or existing application to:

- Manage a collection of updates and inserts to production data
- Version data for “what if” analysis, history, rollbacks

Oracle Workspace Manager: More Information

- **http://otn.oracle.com/products/workspace_mgr**
- **http://www.dbazine.com/fernandez1.html**
  Article in an online magazine highlighting the usefulness of WorkspaceManager to both developers and end users.
- **http://groups.google.com/groups?q=Workspace+Manager&hl=en&lr=&selm=a758oh%24eqv%241%40lust.ihug.co.nz&rnum=7**
  A favorable review in comp.databases.oracle.server

William.Beauregard@oracle.com

Oracle Workspace Manager: Questions From DBA SIG

- Can Workspace Manager be used for auditing?
  - Yes, Workspace Manager can keep track of the versions of changes, the time of the change and the Oracle user that made the change.
- Does Workspace Manager rebuild the primary key index when it enables-versioning or disables-versioning on a table?
  - Yes, in order to uniquely identify each row version, Workspace Manager combines versioning metadata with the primary key. However, the new index works in the same way with existing application SQL DML statements. If versioning is disabled on a table, Workspace Manager rebuilds the index without the versioning metadata.

Oracle Workspace Manager: Questions From DBA SIG

- Can Workspace Manager enforce quotas (limits) on the number of workspaces and versions created?
  - We've not seen a situation yet where the number of workspaces was an issue and the overhead associated with creating a workspace is insignificant. DBAs already have the ability to restrict a user's use of spaceand hence versions through their profile. Therefore, Workspace Manger doesn't enforce quotas. If this is a requirement for a real application using Workspace Manager please let me know.

Oracle Workspace Manager: Questions From DBA SIG

- **Does Workspace Manager support partitioning (to enable row versions to be routed to another physical space)?**
  - **Workspace Manager Release 9.2 does support partitioning. However, if a partitoned index is used and versioning is disabled on a partitioned table, Workspace Manager does not currently rebuild the index. The plan is to release a patch to enable automatic index rebuilding.**
- **Does Workspace Manager support Data Guard (Physical & logical Standby)**
  - **Yes**

Oracle Workspace Manager: Questions From DBA SIG

For further questions about Workspace Manager please contact:

Bill Beauregard
Principal Product Manager

Oracle Corporation
William.Beauregard@oracle.com
One Oracle Drive **office: 603.897.3021**
Nashua, NH 03062 **fax: 603.897.3334**
cell: 617.283.5366