CED Administrator's Guide (v3.0)

By Theo Larrieu

Introduction

Overview

This document addresses topics related to the high-level organization and administration of the CEBAF Element Database (CED) infrastructure. Topics covered include how to create the database and how to move data between workspaces and schemas. This document will evolve as our tools and methods change over time, so be sure to check the CED wiki (https://devweb/twiki/bin/viewfile/AHLA/CED?filename=CED_Administrator.pdf) to make sure you are referencing the most up-to-date version. The multi-step procedures outlined in early versions of this document will assuredly be replaced by simpler procedures that involve the execution of scripts or stored procedures.

Database Creation

Define CED Roles

A prerequisite to creating of a CED schema is to create the READ_CED and OWN_CED roles. When it is created, each CED schema will grant SELECT privilege on all its tables to the READ_CED role and will grant ALL privilege to the OWN_CED role.

Because each CED schema is granted the READ_CED role by default at creation time, each CED schema has the ability to view, but not modify the data from other CED schemas in the same database instance. The OWN_CED role when granted permits that user (e.g. CED3_OWNER) also to insert, update, and delete data across all CED schemas.

```
%> scp devcvs:/cs/dvlhome/apps/c/CED/dvl/src/createCEDRoles.sql  /oracle/scripts
%> cd /oracle/scripts
%> sqlplus '/ as sysdba'

SQL> @createCEDRoles.sql
'*****************************************'
'* NOTE: This script must be run as sysdba *'
'*****************************************'
SQL> exit
%>
```

Create CED Users

A CED Database user is created by connecting to the oracle database as the SYSDBA and running the script createUser.sql script as the example in the textbox below illustrates.

```
%> scp devcvs:/cs/dvlhome/apps/c/CED/dvl/src/createCEDUser.sql  /oracle/scripts
%> cd /oracle/scripts
%> sqlplus '/ as sysdba'

SQL> @createCEDUser.sql
'*****************************************'
'* NOTE: This script must be run as sysdba *'
'*****************************************'
Enter the schema owner to create >ced3_owner
Enter the schema password to set >*******

User created.
Grant succeeded.
(…)
Grant succeeded.
Grant succeeded.
SQL> exit
%>
```

Create CED Schema Objects

To define the tables, views, sequences and other objects in a CED schema, use SQLPLUS to connect to Oracle as the user who owns the schema and execute the create createSchema3.sql script as illustrated in the example below:

```
%> sqlplus ced3_owner@ceddb01

SQL>@/cs/dvlhome/apps/c/CED/dvl/src/createSchema3.sql

SQL> exit
```

The createSchema3.sql script creates empty tables, sequences, and views. In a multiple-schema environment, it must be executed within each schema by the owner of that schema.

The createSchema3.sql is generated programmatically using the ERWin Entity-Relationship Modeling software and should not be hand-edited (hand-edits will be lost the next time the script is generated). Changes should instead be made to the ERWin model and then a new createSchema3.sql script generated from there. The ERWin Model file is stored in the CED_V3.erwin file located in the CED/dvl/doc directory.

Multiple Schemas

While it would be possible to run the CED using a single versioned schema, we have opted to spread the functionality across four schemas as illustrated in both Table 1 and **Error! Reference source not found.** below.

Table 1

Schema	Versioned?	Role	Description
CED3_OWNER	N	OWN_CED	Used by API to connect. Has privilege to modify data in all CED schemas. Owns the CED_ADMIN

			PL/SQL package.
CED3_OPS	N	READ_CED	Current Operational data. For performance reasons, is not versioned
CED3_DEVL*	Y	READ_CED	Contains Multiple Developmental Workspaces. "LIVE" intended to be a mirror of CED3_OPS.
CED3_HIST	Y	READ_CED	Contains Single LIVE workspace to preserve a history of CED3_OPS via creation of regular savepoints.
** We couldn't use the name CED3_DEV here as originally because the 8 character name seemed to tickle an Oracle bug that prevented DBMS_WM.enableVersioning from succeeding.*			

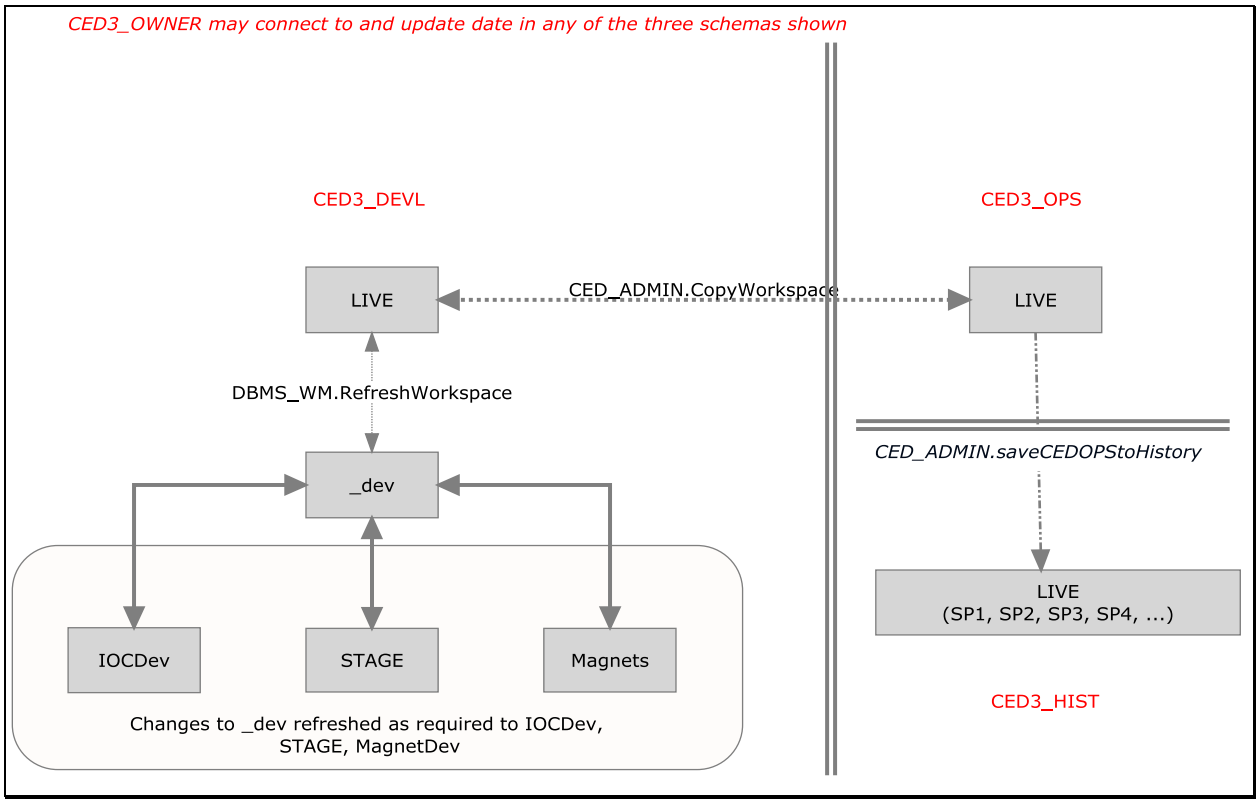

Figure 1

The bundle of four schemas (CED3_OWNER, CED3_OPS, CED3_DEVL, CED3_HIST) will comprise a single production CED installation. When no workspace or savepoint name is specified to make a CED connection, the CED API Library will by default access the contents of the CED3_OPS schema. When a valid workspace name is specified, data will be retrieved from CED_DEVL and when a valid savepoint name is specified, data will be retrieved from CED3_HIST.

Enable Versioning

Two of the CED schemas (CED3_DEVL and CED3_HIST) will be version-enabled using the Oracle Workspace Manager toolkit. This means that the tables in these schemas will be capable of storing multiple versions of each data row in their tables. Rows may be versioned using either workspaces or savepoints or both. (For further discussion of workspace and savepoints, see the Oracle Workspace Manager manual available at http://devweb/oradocs/appdev.111/b28396/toc.htm).

In our usage, the CED3_DEVL is intended to be versioned using workspaces, while the CED3_HIST tables will be versioned using savepoints. In both cases, the enableCEDVersioning script is run to version-enable the tables in each schema.

```
%> sqlplus ced3_devl@ceddb01

SQL>@/cs/dvlhome/apps/c/CED/dvl/src/enableCEDVersioning.sql
SQL> exit

%> sqlplus ced3_hist@ceddb01

SQL>@/cs/dvlhome/apps/c/CED/dvl/src/enableCEDVersioning.sql
SQL> exit
```

After enableCEDVersioning completes, the original tables will have been renamed with an _LT suffix and replaced with a view of the original name as can be seen in Figure 2.

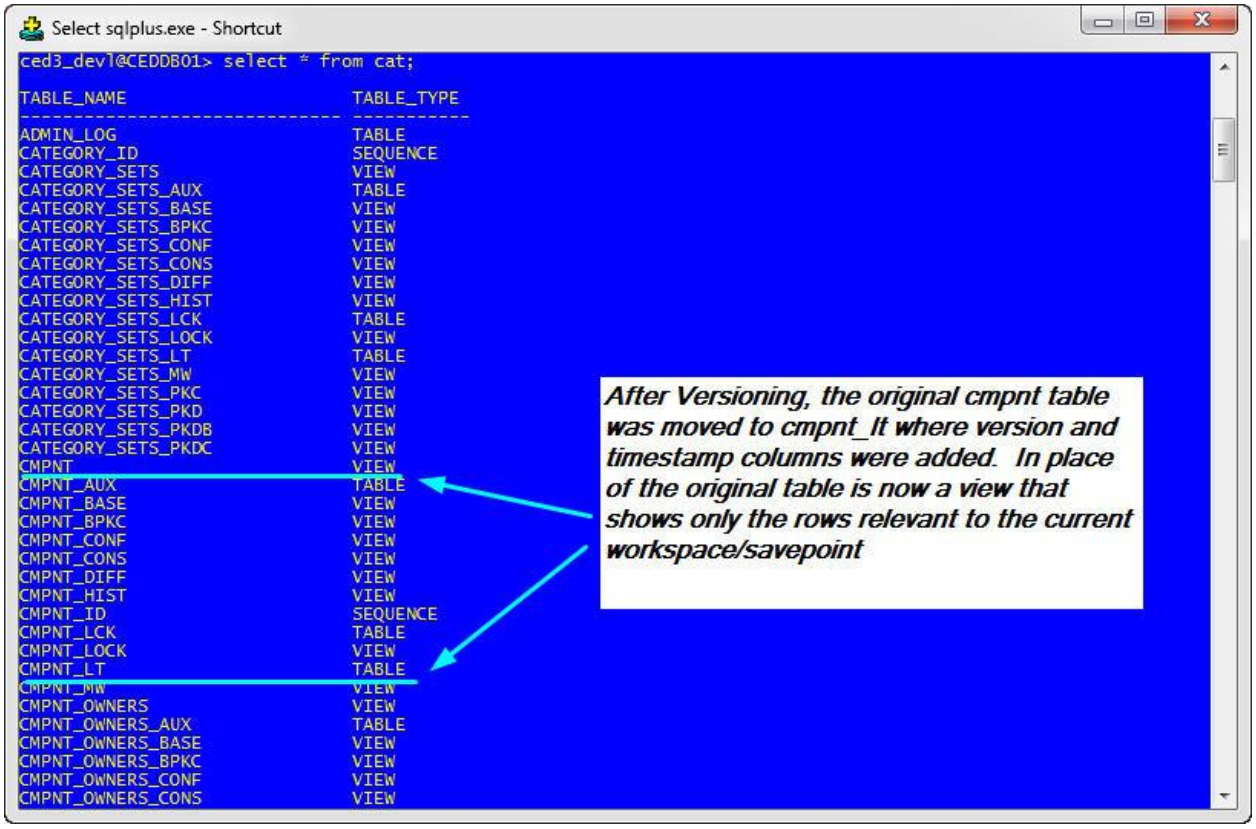

Figure 2

The CED_ADMIN Package

A package of stored procedures named CED_ADMIN facilitates the creation and maintenance of multiple CED schemas. It should be installed in and owned by the CED3_OWNER schema, however all schemas with the READ_CED role should be granted execute permission on it and allowed to access it via a public synonym.

```
%> sqlplus ced3_owner@ceddb01

SQL>@/cs/dvlhome/apps/c/CED/dvl/src/CED_ADMIN.pkg.sql
SQL>/

Package body created.

SQL>grant execute on CED_ADMIN to READ_CED;

Grant succeeded.

SQL>create public synonym CED_ADMIN for CED_ADMIN;

Synonym created.

SQL>grant execute on CED_ADMIN to WRITE_CED;

Grant succeeded.

/* We also need to grant permissions explicitly to certain users because
   Role-based permissions are not honored in some contexts such as when a
   Job runs scheduled via DBMS_JOB. */

SQL>grant execute on CED_ADMIN to CED3_DEVL;

Grant succeeded.

SQL>grant execute on CED_ADMIN to CED3_HIST;

Grant succeeded.

SQL>grant execute on CED_ADMIN to CED3_OPS;

Grant succeeded.

/*  We just need one master copy of the Staff materialized view */

ced3_owner@CEDDB01> grant select on staff to READ_CED;

Grant succeeded.

SQL> exit
```

Tweaks for READ_CED schemas

After creating non-owner schemas (e.g those with READ_CED, not OWN_CED role), the following minor changes should be made to the schema.

```
%> sqlplus ced3_dev1@ceddb01

SQL> create synonym CED_ADMIN for ced3_owner.CED_ADMIN;

Synonym created.

SQL> exec CED_ADMIN.grantPermissions;

PL/SQL procedure successfully completed.

SQL> exec CED_ADMIN.createStageTables;

PL/SQL procedure successfully completed.

SQL> drop table web_auth;

Table dropped.

SQL> create synonym web_auth for ced3_owner.web_auth;

Synonym created.

SQL> drop table admin_lock;

Table dropped.

SQL> create synonym admin_lock for ced3_owner.admin_lock;

Synonym created.

SQL> drop materialized view staff;

Materialized view dropped.

SQL> create synonym staff for ced3_owner.staff;

Synonym created.

SQL> exit
```

Explicit Permissions for Owner Schema

Permissions granted via role are not available in certain circumstances such as when a stored procedure is executed via DBMS_JOB. Therefore in addition to running the createCEDRoles.sql script as described earlier, it is necessary for the DBA to grant some permissions explicitly to the master Schema (i.e. CED3_OWNER)

```
# On the database server
%> sqlplus '/ as sysdba'

SQL> exec DBMS_WM.GrantSystemPriv('ACCESS_ANY_WORKSPACE', 'CED3_OWNER', 'NO');

PL/SQL procedure successfully completed.

SQL> exec dbms_wm.grantsystempriv('MERGE_ANY_WORKSPACE', 'CED3_OWNER', 'NO');

PL/SQL procedure successfully completed.

SQL>exit
```

Explicit Permissions for DEVL Schema

Permissions granted via role are not available in certain circumstances such as when a stored procedure is executed via DBMS_JOB. Therefore in addition to running the createCEDRoles.sql script as described earlier, it is necessary for the DBA to grant some permissions explicitly to the development Schema (i.e. CED3_DEVL)

```
# On the database server
%> sqlplus '/ as sysdba'

SQL> exec DBMS_WM.GrantSystemPriv('ACCESS_ANY_WORKSPACE', 'CED3_DEVL', 'NO');

PL/SQL procedure successfully completed.

SQL> exec dbms_wm.grantsystempriv('MERGE_ANY_WORKSPACE', 'CED3_DEVL', 'NO');

PL/SQL procedure successfully completed.

SQL>exit
```

Create Workspaces

After a schema has been version-enabled, it is possible to create workspaces in it using procedures made available through Oracle Workspace Manger. The following example shows the commands used to create the workspace set shown earlier in **Error! Reference source not found.**.

```
%> sqlplus ced3_devl@ceddb01

SQL> EXEC DBMS_WM.gotoworkspace('LIVE');

/* _dev will be "hidden" parent to development workspaces like IOCDev */
SQL> EXEC DBMS_WM.createWorkspace('_dev');

SQL> EXEC DBMS_WM.gotoWorkspace('_dev');
SQL> EXEC DBMS_WM.createWorkspace('IOCDev',' Workspace for maintaining IOC data');
SQL> EXEC DBMS_WM.createWorkspace('MagnetDev', 'Workspace for adding new Magnets and
related properties');
SQL> EXEC DBMS_WM.createWorkspace('STAGE', 'Workspace for AHLA group to add and edit
bealmine elements and merge Elegant Decks');

SQL> exit
```

- Note that when workspaces are created, they are owned by the user who created them. The API relies on this owner information to determine which schema to use. While it may be correct to create the workspaces as user CED3_OWNER from an Oracle perspective, it will cause the API problems because it will not know to switch to schema CED3_DEVL before performing queries.

Administrative Tasks

Background

Definitions

In the context of Oracle Workspace Manager, to *merge* data in a workspace means to apply changes from a child workspace to its immediate parent in the workspace hierarchy. To *refresh* means the opposite: to apply changes from a parent to its child.

CED Workspace Naming Convention

CED workspaces whose name begins with an underscore character (e.g. _dev) are considered to be administrative workspaces and are not advertised via the API *CEDdb->stages()* function call or the *ced -workspaces* command line invocation. Otherwise, the same rules apply to them as to other workspaces.

Continually refreshed workspaces

A workspace that is continually refreshed will automatically be refreshed when data is committed to or merged into its parent workspace. A workspace may be defined as continually refreshed when it is created or at a later point in time, however once a workspace is designated continually refreshed, that feature may not be disabled without removing and recreating the workspace. It's also important to note that once a row has been modified in a child workspace, that row will no longer receive automatic updates from the parent workspace.

Based on experience, the implementation of continually refreshed workspaces by Oracle causes too much trouble when merging tables with non-primary key uniqueness constraints. As of June 2013, the child workspaces of _dev are no longer set to continually refreshed.

Some useful queries and commands related to continually refreshed workspaces are illustrated below.

```
/* Query to find out which workspaces are continually refreshed */
SQL> select workspace, continually_refreshed from all_workspaces;
WORKSPACE                      CONTINUALLY_REFRESHED
------------------------------ ---
LIVE                           NO
IOCDev                         YES
MagnetDev                      YES
STAGE                          YES
MyDev                          NO
_dev                           NO

/* Make a workspace continually refreshed if it is not already */
SQL> EXECUTE DBMS_WM.ChangeWorkspaceType('MyDev');

/* Specify continually refreshed at Workspace Creation Time */
SQL> EXECUTE DBMS_WM.CreateWorkspace(workspace=>'IOCDev', isRefreshed=>true);
```

Merging a workspace

The instructions in this section deal specifically with the workspaces in the CED3_DEVL schema (see **Error! Reference source not found.**).

Note that in our multiple-schema configuration, it is in fact, the CED3_OPS schema that we treat as the "live" or "production" version of the data. Therefore, **simply merging data into the LIVE workspace of CED3_DEVL does not make it visible to Operational tools**. Making data operational involves an additional step of copying the data to CED3_OPS schema which will be discussed later in this document. It is also important to make sure that the CED3_DEVL LIVE workspace has been updated with any LIVE-EDITS that may have been made in CED3_OPS.

Case 1 - merge STAGE into _dev

Follow these steps which will require access to a terminal on a Control System Linux workstation:

1. Run the ced_audit command to verify the consistency of the STAGE workspace
2. Execute DBMS_WM.mergeWorkspace procedure to merge STAGE into _dev
3. commit or rollback the merge
4. Run the ced_audit command to verify the consistency of the _dev workspace

```
#Audit the STAGE workspace
%>  ced_audit -wrkspc STAGE -f /dev/stdout -e -p

# Connect to sqlplus as ced3_devl
%> sqlplus ced3_owner@ceddb01

/* Now merge the pending changes from STAGE into _dev */
SQL> EXEC DBMS_WM.mergeWorkspace('STAGE', auto_commit=>false);

/* If there was no error, commit the changes, else rollback;
SQL> commit;

/* exit sqlplus */
SQL> exit

#Audit the _dev workspace
%>  ced_audit -wrkscp _dev -f /dev/stdout -e -p
```

Case 2 - merge _dev into LIVE

Follow these steps which will require access to a terminal on a Control System Linux workstation:

1. Run the ced_audit command to verify the consistency of _dev workspace
2. Compare the modify_date timestamps between CED3_DEVL LIVE and CED3_OPS LIVE.
3. If CED3_OPS has a more recent timestamp, it means there have been LIVE-EDITS that must be copied to CED3_DEVL. Run the CED_ADMIN.copyWorkspace procedure to pull those changes into CED3_DEVL LIVE.
4. Execute DBMS_WM.mergeWorkspace procedure to merge _dev into LIVE
5. commit or rollback the merge
6. Run the ced_audit command to verify the consistency of the LIVE workspace

```
#Audit the _dev workspace
%>  ced_audit -wrkspc _dev -f /dev/stdout -e -p

# Connect to sqlplus as ced3_devl
%> sqlplus ced3_devl@ceddb01

/* Copies the OPS Data into current workspace, changing only differing rows */
SQL> EXEC CED_ADMIN.copyWorkspace('LIVE','CED3_OPS');

/* Now merge the pending changes from _dev into LIVE */
SQL> EXEC DBMS_WM.mergeWorkspace('_dev', auto_commit=>false);

/* If there was no error, commit the changes, else rollback;
SQL> commit;

/* exit sqlplus */
SQL> exit

#Audit the LIVE workspace
%>  ced_audit -f /dev/stdout -e -p
```

Case 3 - Selectively merge a single element from IOCDev into _dev

There will be times when it is not desirable or possible to merge all of the changes in a child workspace into its parent. This is frequently the case right now with the IOCDev workspace where a developer may request that a single new IOC be made available to OPS, but it is not desirable to merge the entire IOCDev workspace which also contains multiple incomplete IOCs.

Case 3a - Using the web

The simplest way to merge a single IOC from IOCDev into _dev is to use the web-based merge tool provided as part of the CED web application (Figure 3). Note however, that the web-based merge only merges element property values and is not appropriate if changes have been made to property definitions in the catalog (cmpnt_type_*) tables. The web tool also has no capacity to deal with conflicts. If a conflict prevents the merge from completing successfully, it will be necessary to resort to manual intervention via sqlplus to resolve the conflict.

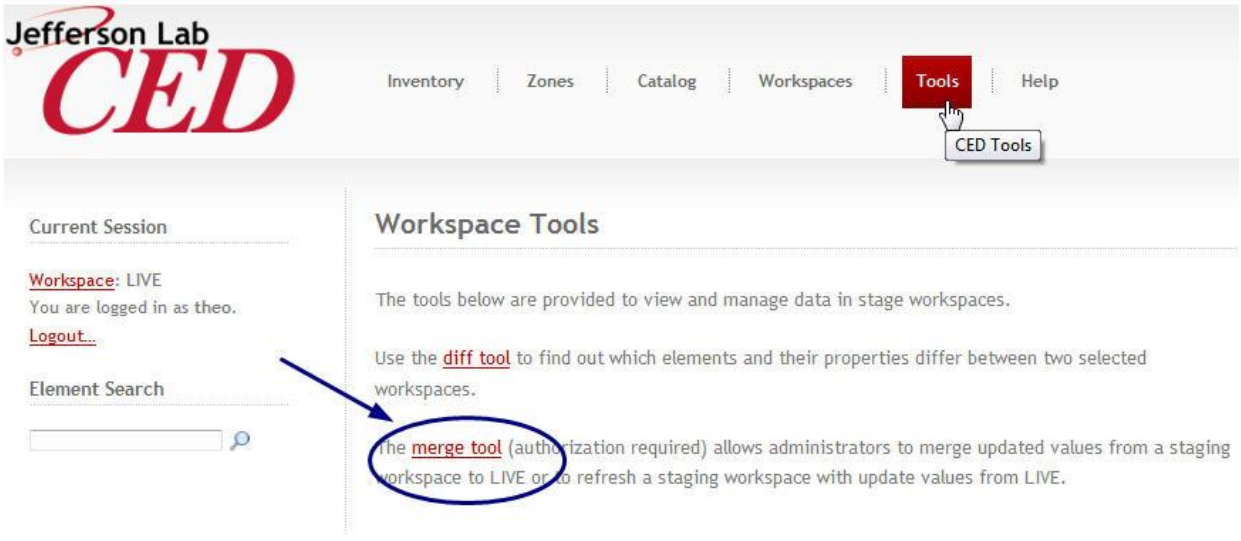

Figure 3 - Accessing the web based merge.

The merge tool requires authentication as a CED administrator and will prompt the user to log in if he/she has not already done so. The next step is to select the child workspace from which elements will be merged using the provided drop-down menu (Figure 4). Pressing Submit will initiate a search for elements and property values that differ between the selected workspace and its parent.

Depending on the number of differences between the child workspace and its parent, the search may take considerable time (as long as 5 minutes perhaps) before the results are displayed in a table on a new page similar to the one illustrated in Figure 5. The table contains a row for each element whose property values differ between the selected workspace (right column) and its parent (left column) as well as the specific properties that differ. There is also a checkbox in each row beside the element name. Select the checkbox in the right-hand column next to each element to be merged and then scroll to the bottom of the table where there is a button labeled "update selected elements". Press this button to initiate the merge process.

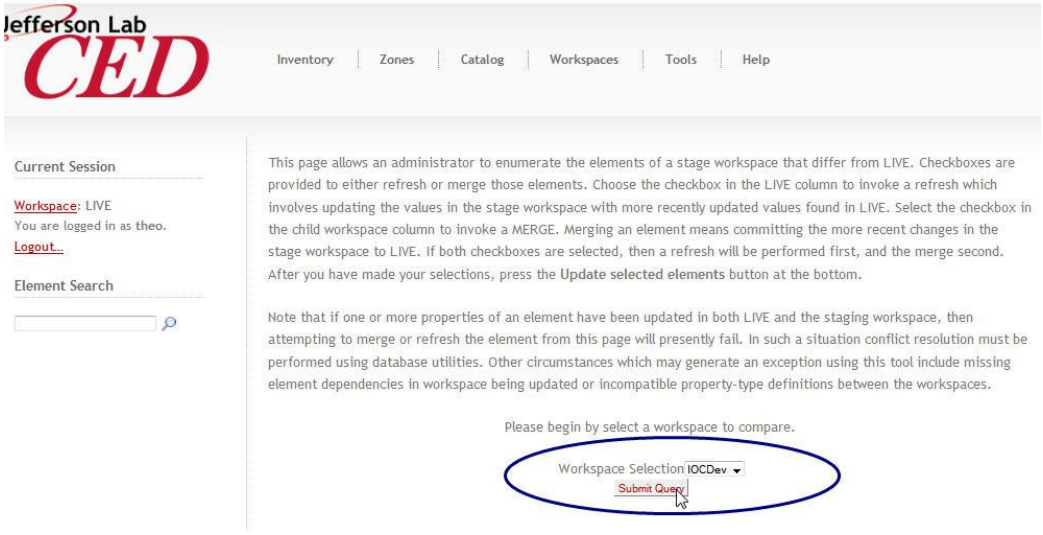

Figure 4

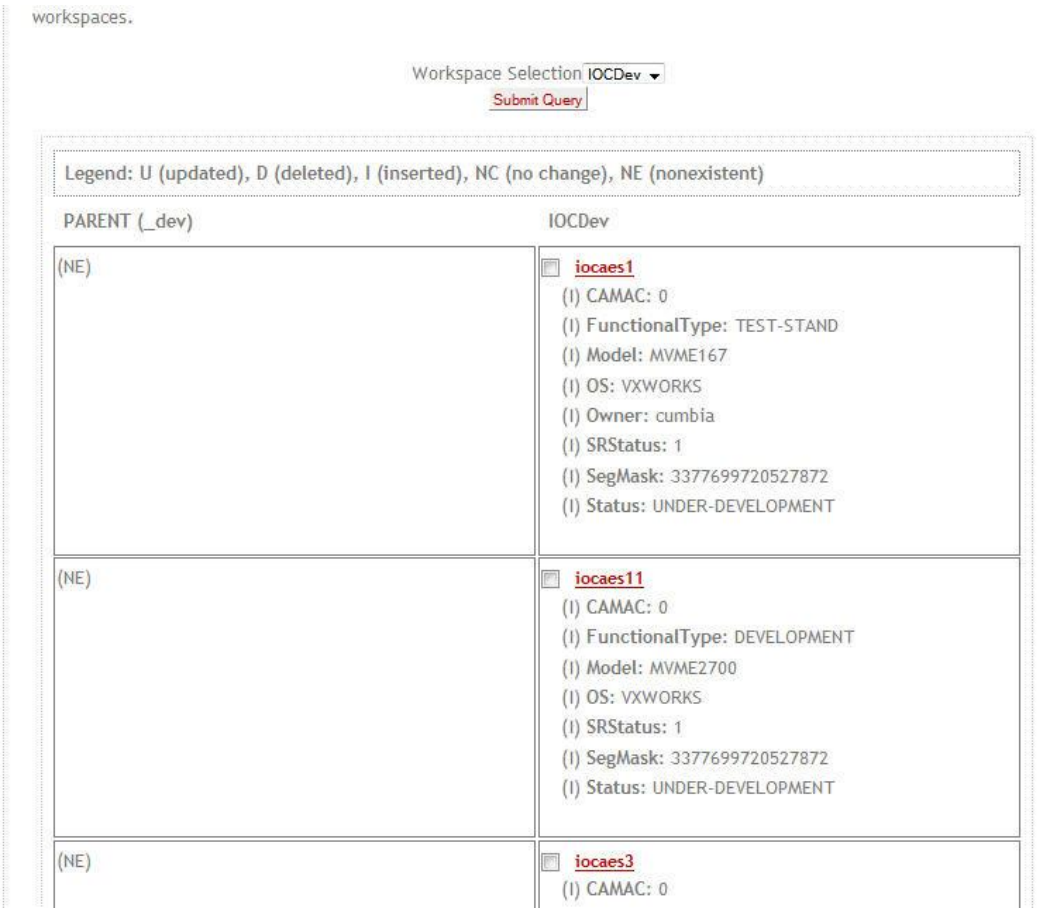

Figure 5

Case 3b - From sqlplus

The web-based merge tool can be slow if the workspace contains numerous changes…

Refreshing a workspace

The instructions in this section deal specifically with the workspaces in the CED3_DEVL schema (see **Error! Reference source not found.**).

Case 1 - Refresh IOCDev with latest CED3_OPS data

```
# Connect to sqlplus as ced3_devl
%> sqlplus ced3_devl@ceddb01

/* Copies the OPS Data into current workspace, changing only differing rows */
SQL> EXEC CED_ADMIN.copyWorkspace('LIVE','CED3_OPS');

/* Now do a refresh to make the copied data available to _dev and its
   Continually refreshed child workspaces (including IOCDev) */
SQL> EXEC DBMS_WM.refreshWorkspace('_dev', auto_commit=>false);

/* If there was no error, commit the changes, else rollback; */
SQL> commit;

/* exit sqlplus */
SQL> exit

# (Optional) Audit the IOCDev workspace
%>  ced_audit -f /dev/stdout -wrkspc IOCDEv -e -p
```

It may be tempting to copy the contents of CED3_OPS.LIVE directly into CED3_DEVL.IOCDev and skip the intermediate steps in the recipe above, but this should be avoided because doing so is likely to result in conflicts the next time IOCDev has to be merged. Resolving the conflicts is a more cumbersome procedure than the extra steps above.

Dealing with conflicts

The following types of conflicts can arise during a merge or refresh operation

- A Workspace Manager conflict can occur if the same row of data has been updated in both the parent and the child workspace.
- A CED conflict can occur if the same element or property has been created independently in two workspaces. The two elements or properties will have different primary keys, but share the same name. Attempting to merge or refresh the two workspaces together will result in an integrity constraint violation because of the duplicated name.

```
/* The following error message was encountered while merging elements via the web page */
EXCEPTION: Failed to merge iocse12BpmCrate from STAGE to _dev.

ORA-20055: conflicts detected for workspace: 'STAGE' in table: 'CED3_DEVL.CMPNT_PROP_FK_CMPNT'
ORA-06512: at "WMSYS.LT", line 7172
ORA-06512: at line 1


ced3_devl@CEDDB01> column WM_WORKSPACE format A12
ced3_devl@CEDDB01> select * from cmpnt_prop_fk_cmpnt_conf;

WM_WORKSPACE   CMPNT_ID      VALUE CMPNT_TYPE_PROP_ID   VALUE_ID       DIM1       DIM2 WM_DEL
------------ ---------- ---------- ------------------ ---------- ---------- ---------- -----
STAGE             14351      16129               2390                     0          0 NO
BASE              14351      14359               2390     313026          0          0 NO
_dev              14351      14359               2390     313026          0          0 YES

/* In this case the property was deleted in _dev and updated in STAGE.  We want to keep the
value from STAGE which is the CHILD of _dev */

SQL> EXECUTE DBMS_WM.BeginResolve ('STAGE');

PL/SQL procedure successfully completed.

SQL> EXECUTE DBMS_WM.ResolveConflicts ('STAGE', 'CMPNT_PROP_FK_CMPNT', 'CMPNT_ID=14351',
'CHILD');

PL/SQL procedure successfully completed.

SQL> COMMIT;

Commit complete.

SQL> EXECUTE DBMS_WM.CommitResolve ('STAGE');

/* Now we can return to the web page and try the merge again and not get the error. */
```

For more discussion of versioning and dealing with conflicts, see the Wiki topic at:
https://devweb/twiki/bin/view/AHLA/CEDVersioningImplementation

Copying data between schemas

Copying Data from CED3_DEVL to CED3_OPS

Once data has been merged into the CED3_DEVL.LIVE workspace, it will often be desired to place the data into CED3_OPS where it will become the new production "OPS" (AKA "LIVE") data.

```
# Connect to sqlplus as ced3_ops
%> sqlplus ced3_ops@ceddb01

/* Copies the OPS Data into current workspace, changing only differing rows */
SQL> EXEC CED_ADMIN.copyWorkspace('LIVE','CED3_DEVL');


/* exit sqlplus */
SQL> exit

# Audit the OPS workspace
%>  ced_audit -f /dev/stdout -e -p
```

Creating a Savepoint of CED3_OPS in CED3_HIST

Because CED3_OPS is not version-enabled for performance reasons, savepoints are instead stored in the CED3_HIST schema which is version-enabled. The process of creating a savepoint therefore entails copying the changes (if any) from CED3_OPS to CED3_HIST and then creating a savepoint in CED3_HIST.

The way to do this is by calling the CED_ADMIN.saveCEDOPStoHistory function. Because the user who invokes the procedure will be the owner of the savepoint, this function should only ever be invoked by the ced3_hist user. (The saveCEDOPStoHistory function has a check to make sure it's being invoked by the correct user, but why tempt fate?)

```
%> sqlplus ced3_hist@ceddb01

SQL> exec CED_ADMIN.saveCEDOPStoHistory;

11-NOV-2011 17:07: Beginning saveCEDOPStoHistory
11-NOV-2011 17:07: Beginning checkoutCEDOPS
11-NOV-2011 17:07: Beginning copyWorkspace (Local)
11-NOV-2011 17:07: create database link CEDSRC
11-NOV-2011 17:07: zones removed.... 0
11-NOV-2011 17:07: zones_links removed.... 0
11-NOV-2011 17:07: segments removed.... 0
11-NOV-2011 17:07: fk_cmpnt values removed.... 0
11-NOV-2011 17:07: cmpnt removed.... 0
11-NOV-2011 17:07: cmpnt removed.... 0
11-NOV-2011 17:07: cmpnt_type removed.... 0
(…)
11-NOV-2011 17:07: CMPNT_PROP_DATE added.... 0
11-NOV-2011 17:07: CMPNT_PROP_FK_CMPNT added.... 0
11-NOV-2011 17:07: CMPNT_PROP_FLOAT added.... 0
11-NOV-2011 17:07: CMPNT_PROP_INTEGER added.... 0
11-NOV-2011 17:07: CMPNT_PROP_STRING added.... 3
11-NOV-2011 17:07: cmpnt_prop_blob added.... 0
11-NOV-2011 17:07: cmpnt_type_owners added.... 0
11-NOV-2011 17:07: cmpnt_type_prop_owners added.... 0
(…)
11-NOV-2011 17:07: cmpnt_prop rows updated... 10
11-NOV-2011 17:07: cmpnt_prop_fk_cmpnt rows updated... 0
11-NOV-2011 17:07: cmpnt_prop_bool rows updated... 1
11-NOV-2011 17:07: cmpnt_prop_integer rows updated... 0
11-NOV-2011 17:07: cmpnt_prop_float rows updated... 1
11-NOV-2011 17:07: cmpnt_prop_string rows updated... 7
11-NOV-2011 17:07: cmpnt_prop_date rows updated... 0
11-NOV-2011 17:07: cmpnt_prop_blob rows updated... 0
11-NOV-2011 17:07: Beginning compareRowCounts ....
11-NOV-2011 17:07: MATCH: ZONES...SRC = 83 DEST = 83
11-NOV-2011 17:07: MATCH: SEGMENTS...SRC = 53 DEST = 53
11-NOV-2011 17:07: MATCH: ZONE_LINKS...SRC = 160 DEST = 160
11-NOV-2011 17:07: MATCH: CMPNT_PROP_FK_CMPNT...SRC = 8910 DEST = 8910
(…)
11-NOV-2011 17:07: MATCH: CMPNT_TYPE_PROP_DEF...SRC = 376 DEST = 376
11-NOV-2011 17:07: MATCH: CMPNT_TYPE_PROP_DIM...SRC = 338 DEST = 338
11-NOV-2011 17:07: MATCH: CMPNT_TYPE_PROP_REQ...SRC = 317 DEST = 317
11-NOV-2011 17:07: MATCH: CMPNT_TYPE_PROP_OWNERS...SRC = 0 DEST = 0
11-NOV-2011 17:07: End copyWorkspace
11-NOV-2011 17:07: Finished checkoutCEDOPS
11-NOV-2011 17:07: Create Savepoint AutoSP4 - Automatically generated savepoint
11-NOV-2011 17:07: Finished saveCEDOPStoHistory

PL/SQL procedure successfully completed.
```

Automated Nightly Savepoint

On dboracle which hosts the ceddb01 database, there is a crontab entry for the oracle user that executes the script */app/oracle/scripts/CEDhistsave.ksh* every night. This shell script calls the stored procedure CED_ADMIN.SaveCEDOPStoHistory to create a savepoint.

Restoring CED3_OPS from a savepoint in CED3_HIST

If it becomes necessary to replace the current data in OPS with data from an historical savepoint, this can be done using the CED_ADMIN package routine restoreCEDOPSFromHistory.

```
%> sqlplus ced3_ops@ceddb01

-- List out the savepoints if necessary

SQL> select savepoint, createtime, description from all_workspace_savepoints where owner='CED3_HIST' order by createtime;

 (…)
AutoSP77                  13-JUL-13
Automatically generated savepoint

AutoSP78                  05-AUG-13
Automatically generated savepoint

71 rows selected.

-- Let's presume we want to restore AutoSP77 from the July 13

SQL> exec CED_ADMIN.restoreCEDOPSFromHistory('AutoSP77');

PL/SQL procedure successfully completed.

-- Look at the admin_log table if you want to see more details about the operation:

SQL> select * from admin_log where logdate > sysdate -1 order by logdate;
```

Making Schema Changes

Oracle workspace manager places limits on the type of DDL that may be performed. See the Oracle workspace Manager documentation for details.
http://devweb/oradocs/appdev.111/b28396/long_intro.htm#i1009427

Exporting and Cloning

To CEDTEST for Development

It is often useful to make a copy of the production CED that can be used for testing, development, and debugging. The steps to do so are documented in the SysAdmin Wiki at https://devweb/twiki/bin/view/SysAdmin/HowToCloneProductionCEDDatabase.

To CEDSTBY for Standby

The process of creating the standby database is very similar to cloning for development purposes. The procedure for doing so can be found in the SysAdmin Wiki at
https://devweb/twiki/bin/view/SysAdmin/CEDStandbyDatabaseCreation

Disaster Recovery

Starting up CEDDB01 on dbs

In the event of a sustained outage of dbo where the primary CEDDB01 database is hosted, the CED database can be brought online quicklin on the dbs server using the procedure found in the Sysadmin Wiki at:
https://devweb/twiki/bin/view/SysAdmin/HowToRestoreOracleOnStandbyServer

Web Server Configuration

This section documents configuration of the apache web server and the php apache module to support the CED Web application.

_CED.conf

The following directives are placed in the file /etc/httpd/conf.d/_CED.conf. The file is prefixed with an underscore because apache applies the *.conf files in ascii order.

```
RewriteEngine On
RewriteRule /CSUEApps/c/CED/dvl/wbin/((elem|inventory|edit|new|catalog)/.*)
/CSUEApps/c/CED/dvl/wbin/index.php?url=$1 [PT,QSA]
```

vhosts.conf

The following directives are placed in the file /etc/httpd/conf.d/vhosts.conf on opsweb to support the ced virtual web server. This is what allows us to access the CED as http://ced/ in lieu of http://opsweb/CSUEAPPs/CED/

```
<VirtualHost  *:80>
   ServerName ced
   ServerAlias ced.acc.jlab.org
   DocumentRoot /cs/prohome/apps/c/CED/pro/wbin/
   ErrorLog /var/log/httpd/error_log_ced
   TransferLog /var/log/httpd/access_log_ced
   RewriteEngine On
   RewriteRule /((elem|inventory|edit|new|catalog|zone)/.*) /index.php?url=$1 [PT,QSA]
</VirtualHost>
```

PHP Configuration

ini file

The following directives are placed in the file /usr/csite/pubtools/php/lib/php-web.ini

```
extension=cedlib.so
```

Symbolic link to cedlib.so

```
# Note that no-debug-non-zts-20060613 in the path below is dependent
# on the Zend Engine version of the PHP library.  It will vary between releases.

%> cd /usr/csite/pubtools/php/lib/php/extensions/no-debug-non-zts-20060613/

# Make a symbolic link to the desired PHP version of cedlib
%>ln -s /cs/certified/apps/cedlib/lib/rhel-6-ia32/php3.0/cedlib.so ./cedlib.so
```

CED Server external proc config

.so file

Is placed in $ORACLE_HOME/lib on the database server.

Extproc.ora

Beginning with Oracle 11.2.0, the following directives (in bold) are placed in the file $ORACLE_HOME/hs/admin/extproc.ora instead of configuring listener.ora as was done in earlier versions of Oracle.

```
SET EXTPROC_DLLS=ANY
```

Listener.ora

In versions of Oracle prior to 11.2.0, the following directives (in bold) are placed in the file $ORACLE_HOME/network/admin/listener.ora

```
/* The directives below are only for versions of Oracle prior to 11.2 */

SID_LIST_LISTENER =
  (SID_LIST =
        (SID_DESC =
                (GLOBAL_DBNAME = CEDSTBY.ACC.JLAB.ORG)
                (ORACLE_HOME = /u01/app/oracle/product/11.1.0/db_1)
                (SID_NAME = cedstby)
        )
        (SID_DESC =
                (PROGRAM = extproc)
                (ORACLE_HOME = /u01/app/oracle/product/11.1.0/db_1)
                (SID_NAME = ced_notify)
        )
```

tnsnames.ora

In versions of Oracle prior to 11.2.0, the following directive is placed in the file $ORACLE_HOME/network/admin/tnsnames.ora

```
EXTPROC_CED_NOTIFY =
  (DESCRIPTION =
     (ADDRESS_LIST =
       (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))
     )
    (CONNECT_DATA =
       (SID=CED_NOTIFY)
       (PRESENTATION = RO)
    )
)
```

Wrapper function

```
# Solaris
SQL> create library ced_notify_lib is
            '/u01/app/oracle/sesrv/11.2.0/db_1/lib/ced_notify.so';
/

# Linux
SQL> create library ced_notify_lib is
            '/opt/oracle/sesrv/11.2.0/db_3/lib/ced_notify.so';
/


CREATE OR REPLACE FUNCTION ced_notify(ced_event IN char) RETURN binary_integer
  AS EXTERNAL
     NAME "ced_notify"
     LIBRARY ced_notify_lib
     LANGUAGE C
     PARAMETERS (ced_event string);
/
```

History Triggers

In order for history triggers in the ced3_devl and ced3_ops schemas to work properly, these schemas must be granted permissions directly (not just via role) to the hist_* tables in the ced3_owner schema.

```
ced3_owner > grant insert, select on hist_cmpnt to ced3_ops, ced3_devl;

Grant succeeded.

ced3_owner> grant insert, select on hist_cmpnt_prop to ced3_ops, ced3_devl;

Grant succeeded.

ced3_owner> grant insert, select on hist_cmpnt_prop_val to ced3_ops, ced3_devl;

Grant succeeded.

ced3_owner> grant select on event_id to ced3_ops, ced3_devl;

Grant succeeded.

ced3_owner> grant execute on ced_notify to ced3_ops, ced3_devl;

Grant succeeded.

ced3_owner> grant execute on CED_ADMIN to ced3_ops, ced3_devl;

Grant succeeded.
```

After the afore-mentioned permissions have been granted, execute the script hist_triggers.sql as ced3_ops. The script can also be excuted for ced3_devl if versioning has not yet been enabled. If versioning has been enabled, then the triggers must be added using the Oracle Workspace Manager beginDDL/comittDDL process.

Logoff Triggers

The trigger below is responsible for calling the event server at the end of a session. It must be installed in each schema.

```
/*
-- Check for invalid triggers
select object_name, object_type, status
from dba_objects
where object_name = 'CED_LOGOFF';
*/


-- as each ced user
create or replace trigger ced_logoff
before logoff on schema
begin
    insert into admin_log (log_id, logdate, logmsg) values (log_id.nextval, sysdate,
'Firing Logoff trigger');
    CED_ADMIN.endHistEvent(true);
    -- insert into admin_log (log_id, logdate, logmsg) values
(ced3_owner.log_id.nextval, sysdate, 'logoff 2 '||to_char(retVal));

end;
/
```