How To Use Data Abstraction Best Practices Privilege Scripts

An Open Source Asset for use with TIBCO® Data Virtualization

|  |  |
| --- | --- |
| **Project Name** | AS Assets Data Abstraction Best Practices |
| **Document Location** | This document is only valid on the day it was printed. The source of the document will be found in the ASAssets\_DataAbstractionBestPractices folder (https://github.com/TIBCOSoftware) |
| **Purpose** | Self-paced instructional |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 8.1.8 | 05/24/2017 | Mike Tinius | Updated for Best Practices v8.1.8 – added Privilege scripts. |
| 8.1.9 | 12/06/2017 | Mike Tinius | Transitioned to Tibco 8.1.9 |
| 2018Q1 | 03/20/2018 | Mike Tinius | Release 2018Q1 – no changes. |
| 2019Q1 | 01/25/2019 | Mike Tinius | Release 2019Q1 - no changes. |

Related Documents

|  |  |
| --- | --- |
| **Name** | **Version** |
| How To Use Utilities.pdf | 2018Q1 |
| How To Use Data Abstraction Best Practices.pdf | 2019Q1 |
| How To Learn Data Abstraction Best Practices.pdf | 2019Q1 |
| How To Test Data Abstraction Best Practices.pdf | 2019Q1 |
| How To Use Data Abstraction Best Practices Manage Annotations.pdf | 2019Q1 |

Supported Versions

|  |  |
| --- | --- |
| **Name** | **Version** |
| TIBCO® Data Virtualization | 7.0 or later |
| AS Assets Utilities open source | 2018Q1 or later |

**Table of Contents**

1 Introduction 5

Purpose 5

Audience 6

References 6

DV Folder Structure 6

Pre-Requisites 9

2 Configuration 10

How to Configure 10

Privilege Script Configuration Summary 10

Best Practices Installation 10

3 Executing Resource Privileges 13

Introduction 13

Instructions 13

4 Privilege Spreadsheet and Database Table Mappings 15

Spreadsheet “sheet definition” and Associated Database Table 15

5 Privilege Scripts Method Definition 18

Detailed Definitions 18

Privilege Maintenance: Get Privileges 18

1. getPrivileges 18

Privilege Maintenance: Update Privileges 20

2. updatePrivilegesDriver 20

Privilege Maintenance: Validate Privileges 23

3. validatePrivilegesDriver 23

Group Maintenance: Get Groups 26

4. getGroups 26

Group Maintenance: Update Groups 27

5. updateGroupsDriver 27

Group Maintenance: Delete Groups 29

6. deleteGroupsDriver 29

Group Maintenance: Validate Groups 31

7. validateGroupsDriver 31

Group Maintenance: Get Users 33

8. getUsers 33

User Maintenance: Update Users 34

9. updateUsersDriver 34

User Maintenance: Delete Users 35

10. deleteUsersDriver 35

User Maintenance: Validate Users 37

11. validateUsersDrivers 37

Database Maintenance: Load Database Privileges 38

12. loadPrivileges\_EXCEL 38

Database Maintenance: Load Database Groups 39

13. loadGroups\_EXCEL 39

Database Maintenance: Load Database Users 41

14. loadUsers\_EXCEL 41

Database Maintenance: Perform Supporting Table Maintenance 42

15. performSupportingTableMaintenance 42

Database Maintenance: Fix Privilege Table 44

16. fix\_PROD\_ORA\_COMPOSITE\_PRIVILEGES 44

16.1. fix\_PROD\_SS\_COMPOSITE\_PRIVILEGES 44

Database Maintenance: Modify Organization Type 45

17. modify\_01\_ORGANIZATION\_TYPE 45

Database Maintenance: Modify Domain Type 46

18. modify\_02\_DOMAIN\_TYPE 46

Database Maintenance: Modify Environment Type 47

19. modify\_03\_ENVIRONMENT\_TYPE 47

Database Maintenance: Modify Project Type 48

20. modify\_04\_PROJECT\_TYPE 48

Database Maintenance: Modify SubProject Type 49

21. modify\_05\_SUBPROJECT\_TYPE 49

Database Maintenance: Modify Composite Privileges 50

22. modify\_06\_COMPOSITE\_PRIVILEGES 50

Database Maintenance: Modify Composite Studio Rights 54

23. modify\_07\_COMPOSITE\_STUDIO\_RIGHTS 54

Database Maintenance: Modify Composite User 56

24. modify\_08\_COMPOSITE\_USER 56

Group Clean-up: Remove Groups 58

25. remove\_groups 58

Group Clean-up: Validate Groups 59

26. validate\_groups 59

1. Introduction

## Purpose

The purpose of Best Practices Privilege Scripts is to provide a framework for performing mass-updates on privileges based on rows maintained in a spreadsheet or a database.

The Privilege Scripts help the developer to maintain and execute privileges during deployment for the various layers. The three main areas for setting privileges are as follows:

/services/databases

/services/webservices

/shared

The strategy for setting privileges fits in generically with any folder structure within the Data Virtualization (DV) server but aligns nicely with the Data Abstraction Best Practices which uses the concept of an “organization” project folder in the three areas above. For example, it might look like this:

/services/databases/My\_Organization

/services/webservices/My\_Organization

/shared/My\_Organization



Figure one: Technical Data Abstraction Layers

## Audience

This document is intended to provide guidance for the following users:

* Data Virtualization Administrators – provides a guide for installation.
* Architects – provides the data abstraction architecture.
* Data professionals – provides background on the published views and usage.
* Operations users – provides insight into triggers and procedures that are executed.
* Project Managers – provides general information on data abstraction best practices.

## References

Product references are shown below. Any references to CIS or DV refer to the current TIBCO® Data Virtualization.

* TIBCO® Data Virtualization was formerly known as
  + Cisco Data Virtualization (DV)
  + Composite Information Server (CIS)

## DV Folder Structure

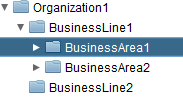
1. DV Folder Structure – establish the baseline concept with “Organization1” as an example.
   1. Databases – Publish views/procedures to a virtual database contained under: /services/databases/Organization1
   2. Web Services – Publish views/procedures to a virtual web service contained under: /services/webservices/Organization1
   3. Shared – Developers for Servicing create resources under: /shared/Organization1
   4. Organization – /shared/Organization1



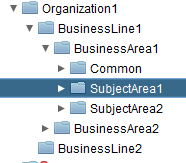
* + 1. Business Line – /shared/Organization1/BusinessLine1



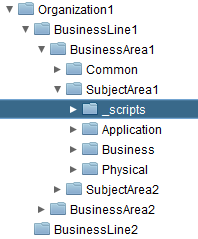
* + - 1. Business Areas – /shared/Organization1/BusinessLine1\BusinessArea1



* + - * 1. Subject Areas – shared/Organization1/BusinessLine1\BusinessArea1\SubjectArea1



Data Abstraction Folder Best Practices



* 1. Data Abstraction Best Practices Folders are contained under each subject area folder.
     1. /Application
     2. /Business
     3. /Physical

1. Privileges on Folders – LDAP groups are created down to the granularity of the subject areas.
   1. /BusinessLine1/BusinessArea1
      1. /Common
      2. /SubjectArea1 – subject area 1 folder
      3. /SubjectArea2 – subject area 2 folder
2. Roles and Rules – very much pattern based in terms of the conventions for creating LDAP names
   1. Admin - **DV\_<BusLine>\_Admin**
      1. Administrator for Servicing folders
   2. Architect - **DV\_<BusLine>\_<BusArea>\_Arch**
      1. Architects are responsible for their own business area /Common and /Servicing/Common. They can promote resources to a /Common folder.
      2. Architects have “G”rant capability.
      3. Architects are responsible for published resources to virtual databases and web services.
   3. Developer - **DV\_<BusLine>\_<BusArea>\_Dev**
      1. Developers are responsible for developing resources in their business area folders. They can create cache objects and execute.
   4. QA - **DV\_<BusLine>\_<BusArea>\_QA**
      1. QA technicians are responsible for executing tests and can use Studio in the QA environment. They are not allowed to modify resources.
   5. Application Id - **DV\_<BusLine>\_<BusArea>\_UserRO\_AppId**
      1. This is a read-only user for applications such as Cognos to connect to Composite.
      2. This user does not have Studio rights.
   6. Read-only user - **DV\_<BusLine>\_<BusArea>\_<SubjArea>\_UserRO**
      1. This is a read-only user for adhoc usage such as Toad users to connect to Composite.
      2. This user does not have Studio rights.
   7. Read-Write user - **DV\_<BusLine>\_<BusArea>\_<SubjArea>\_UserRW**
      1. This is a read-write user for adhoc usage such as Toad users to connect to Composite.
      2. This user does not have Studio rights.

## Pre-Requisites

Follow the steps below to create a new project.

1. LDAP groups have been brought into the target environment.
2. /shared/ASAssets/Utilities have been installed and configured for 2018Q1.
3. Configuration

## How to Configure

This section provides information on how to configure the Privilege Scripts.

### Privilege Script Configuration Summary

1. Copy privilege spreadsheet to file system
2. Modify datasource connections
3. Create privilege database tables
4. Reintrospect datasources
5. Test datasources

### Best Practices Installation

1. Install Best Practices Spreadsheets
   1. Follow the Data Abstraction Best Practices installation directions found in this document: “**How To Use AS Data Abstraction Best Practices.pdf”**
2. Copy privilege spreadsheet to file system
   1. This should have been done during installation of the Best Practices. Verify this.
3. Modify the default settings:
   1. Location: /shared/ASAssets/BestPractices\_v81/\_ProjectMaintenance/defaultValues
   2. datasource: determine whether to use EXCEL or DB\_[LLE,PROD] (database) as the default source for the privilege rows. Value: EXCEL
   3. defaultDatabaseLLEMapping: provides a way to choose Oracle or SQL Server as the default LLE when using "DB\_LLE" datasource. Value: DB\_LLE\_ORA
   4. defaultDatabasePRODMapping: provides a way to choose Oracle or SQL Server as the default LLE when using "DB\_PROD" datasource. Value: DB\_PROD\_ORA
4. Modify the data source path or connection information:
   1. EXCEL: modify the root path if needed.
      1. DV Location: [EXCEL] /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/Privileges\_DS\_EXCEL
      2. E.g. root path: C:/CIS7.0/BestPractices/Privileges
   2. DATABASE: modify connection information if needed for the lower-level environment
      1. Note: This capability allows for a database in LLE for development and testing of privileges and a production database. In reality, a project really only needs a production database. All privileges are considered “production” no matter what DV environment they are being applied to. Therefore, while the scripts for LLE are provided it is not necessary to maintain an LLE privilege database.
      2. Enable either Oracle or SQL Server LLE and/or PROD as needed.
      3. Oracle database locations:

/shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/Privileges\_DB\_PROD\_ORA

/shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/Privileges\_DB\_LLE\_ORA

* + 1. SQL Server database locations:

/shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/Privileges\_DB\_PROD\_SS

/shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/Privileges\_DB\_LLE\_SS

1. Create the tables
   1. Oracle [LLE]: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/DDL/Oracle\_LLE/

1\_pqCreate\_DABP\_privilege\_tables ( CMPPRV, CMPPRV\_DATA )

2\_pqCreate\_DABP\_privilege\_triggers ( CMPPRV )

3\_pqCreate\_DABP\_privilege\_sequences ( CMPPRV )

* 1. Oracle [PROD]: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/DDL/Oracle\_PROD/

1\_pqCreate\_DABP\_privilege\_tables ( CMPPRV, CMPPRV\_DATA )

2\_pqCreate\_DABP\_privilege\_triggers ( CMPPRV )

3\_pqCreate\_DABP\_privilege\_sequences ( CMPPRV )

* 1. SQL Server [LLE]: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/DDL/SQLServer\_LLE/

1\_pqCreate\_DABP\_privilege\_tables ( dbo )

2\_pqCreate\_DABP\_privilege\_triggers ( dbo )

* 1. SQL Server [PROD]: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/DDL/SQLServer\_PROD/

1\_pqCreate\_DABP\_privilege\_tables ( dbo )

2\_pqCreate\_DABP\_privilege\_triggers (dbo )

1. Reintrospect the data sources
2. Test the data sources
   1. Show Contents for one of the Excel data source worksheets to ensure that data is being retrieved.
   2. Show Contents for one of the tables in the chosen data source.
3. Construct the Privilege Spreadsheet to be loaded into the database
4. Load the database from the spreadsheet
   1. loadPrivileges\_EXCEL - the first time, it will notify you of the missing support table values. This will load the privileges from the 3 privilege tabs in the spreadsheet.
   2. performSupportingTableMaintenance - run this to load the supporting tables.
   3. loadGroups\_EXCEL - optional unless you want to set the Studio Rights for the DV groups in the Group\_List tab of the spreadsheet.
   4. loadUsers\_EXCEL – optional
5. Execute the update Groups, Privileges, and Users as needed.

Go to the next section “Executing Resource Privileges” for more information.

1. Executing Resource Privileges

## Introduction

This section provides guidance on how to make a request to the administrator to update the privileges using the manual execution of privileges.

### Instructions

1. Run after hours or early in the morning so as not to impact Developers. Since these scripts are updating the Composite Metadata repository via the API, Developers will notice that their Studios will be unresponsive while these scripts are running.
2. Depending on activity in the network, activity on the Composite server and activity in the Composite repository database server, these scripts may take hours to run.
   1. Update Groups does not take much time at all. This is a very low impact operation.
   2. Update Resource Privileges will take the longest especially depending on the number of privileges to apply.
3. **Update Group Studio Rights before updating privileges**
   1. Sheet [When Datasource=EXCEL]: Group\_List
   2. Table [When Datasource=DB\_PROD or DB\_LLE]: COMPOSITE\_STUDIO\_RIGHTS
   3. Script: **updateGroupsDriver**
      1. **Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/updateGroupsDriver**
      2. Datasource: [DB\_PROD|DB\_LLE|EXCEL] – Indicates which datasource to use to execute from.
      3. *Environment\_Name*: [DEV|**UAT**|PROD**]**
      4. *Organization:* ORG1
      5. ***Project*: Project1**
      6. ***Subproject SB1***
      7. All other parameters are left null
      8. *inDebug1*: Y
      9. *inDebug1Console*: Y
      10. *inDebug1CISLog*: N
      11. *inDebug1ReadOnly*: Y
      12. **Repeat for these projects**:

Project:

Project2

1. **Update Privileges**
   1. Sheet [When Datasource=EXCEL]: Privileges\_shared + Privileges\_databases + Privileges\_webservices
   2. Table [When Datasource=DB\_PROD or DB\_LLE]: COMPOSITE\_PRIVILEGES
   3. Script: **updatePrivilegesDriver**
      1. **Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/updatePrivilegesDriver**
      2. Datasource: [DB\_PROD|DB\_LLE|EXCEL] – Indicates which datasource to use to execute from.
      3. *Batch\_Privileges*: 1
      4. *Environment\_Name*: [DEV|**UAT**|PROD**]**
      5. *Organization:* ORG1
      6. *Project*: **Project1**
      7. *Subproject:* SB1
      8. All other parameters are left null
      9. *inDebug1*: Y
      10. *inDebug1Console*: Y
      11. *inDebug1CISLog*: N
      12. *inDebug1ReadOnly*: Y
      13. *inDebug1RevokeAll*: Y
      14. *inDebug2*: N
      15. *inBypassErrors:* Y
      16. **Repeat for these projects**:

Project: Subproject:

Project2 SB1|SB2|SB3

1. Privilege Spreadsheet and Database Table Mappings

## Spreadsheet “sheet definition” and Associated Database Table

1. Composition of the spreadsheet / database table
   1. Privileges\_shared – This sheet is used to establish privileges for a project and its /shared resources.
      1. Correlates to the database table COMPOSITE\_PRIVILEGES where PRIVILEGE\_TYPE = ‘SHARED’.
   2. Privileges\_databases – This sheet is used to establish privileges for a project and its virtual database resource located at /services/databases.
      1. Correlates to the database table COMPOSITE\_PRIVILEGES where PRIVILEGE\_TYPE = ‘DATABASES’.
   3. Privileges\_webservices – This sheet is used to establish privileges for a project and its virtual web service resources located at /services/webservices.
      1. Correlates to the database table COMPOSITE\_PRIVILEGES where PRIVILEGE\_TYPE = ‘WEBSERVICES’.
   4. Group\_List – The group list sheet is used for applying studio access rights to a group.
      1. Correlates to the database table COMPOSITE\_STUDIO\_RIGHTS.
   5. User\_List – The user list sheet is used for creating composite users that “mirror” the ldap groups for testing purposes.
      1. Correlates to the database table COMPOSITE\_USER.
   6. Roles\_and\_Responsibilities – not used for any scripts (documentation only)
      1. Correlates to the database table ROLE\_TYPE.
   7. Privilege\_Template – not used for any scripts (documentation only)
      1. Correlates to the database table COMPOSITE\_ROLE\_TEMPLATE.
2. Spreadsheet Concepts
   1. Applying privileges
      1. Apply privileges to lowest level folders first. For example, given /shared/folder1 and /shared/folder1/folder2, apply privileges to /shared/folder1 before /folder2.
      2. READ only privileges – The update privileges have the ability to automatically set READ on parent folders. The benefit of this is that these rows do not have to be specified in the spreadsheet any longer like they did with version 1.
      3. Apply a concept of initializing folders to “NONE” recursively first to create a blank pallet on which to paint privileges.
      4. If a folder shares privileges with another sibling folder then don’t apply privileges recursively. Simply set privileges on that folder for your groups. Example:
         1. /shared/folder1 (your folder)
            1. Set recursive mode=”N” for “not” recursive.
         2. /shared/folder2 (another groups folder)
      5. Spreadsheet column: Recurse Child

Recurse Child Resources and Folders:

N or blank = do not recurse. Set specified privileges for the resource only.

YM = Only apply modification recursively (always recommended when using the spreadsheet)

YC = Make child resources look like this resource recursively. This feature is **not recommended** when using the spreadsheet and making multiple modifications because the entire context is not taken into consideration like it does in studio.

Concept 1: Only select YM to recurse child resources when the resource being set is at the lowest level at which it is safe to recursively set privileges for a given USER or GROUP. If the resource is an intermediate level folder with other branches offshoot from it then only apply the privilege modification to the resource folder using "N" unless all branches are required to have the same privilege settings. "YC" should only be used when you want to reset all privileges for other USERS or GROUPS to "NONE" except for the current USER or GROUP being set.

Concept 2: If the higher-level resource folder is set with "YC" then it is not necessary to set this USER or GROUP for branch-level resources.

**Caveat**: It has been determined through testing because the Composite API is being used to set privileges, only “YM” should be used since each transaction on the spreadsheet is executed individually. This differs from the concept in Studio, where Studio has a holistic view of all privileges for all groups for a given resource. Therefore, in Studio, you can set the equivalent of “YC” which is recurse all children and make them look exactly like the parent. Unfortunately, that concept is not the same when executed from this spreadsheet and API.

* 1. Batching privileges
     1. Privileges are batched and executed as a single unit request to DV. The batches are based on finding a set of rows in the spreadsheet with the same path, type and recurse child settings. If any of those change, then that delineates the batch request to DV.
  2. Spreadsheet Line
     1. Each spreadsheet line (like a database row) needs to include all of the information to be able to set a row independently if requested. However, if batching privileges is requested then the first row of a batch is used to set the resource path, type, recurse child.
     2. If the resource path or type are missing, that line in the spreadsheet will \*NOT\* be applied.
     3. It is important to note that the first row of a batch is used exclusively to set the owner and owner domain since this can only be set at the resource path level. By definition of a batch [resource path, type, recurse child] will start a new batch.

1. Privilege Scripts Method Definition

## Detailed Definitions

Detailed documentation on the inputs and outputs can be found in the header and annotation section of each procedure.

1. Composite groups and users – If you are adding groups and users, you must first execute the “updateGroupsDriver” followed by the “udpateUsersDriver”. Groups must be present before users are created because users are assigned to groups upon creation of the user.
2. Validation – you can validate the groups and users were created by using “validateGroupsDriver”, “validateUsersDriver” and “validatePrivilegesDriver”.
3. Delete groups and users – If you want to clean up the composite created users and groups perform these tasks in reverse order of creation. First execute “deleteUsersDriver” to remove the users followed by “deleteGroupsDriver” to remove the groups.

## Privilege Maintenance: Get Privileges

1. getPrivileges **–** Construct a SQL statement based on filters passed in and retrieve the set of privileges that match the criteria.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/getPrivileges

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **Sheet\_Name** – OPTIONAL FILTER: The name of the Excel spreadsheet in which to get/update privileges from: [shared, databases, webservices] | VARCHAR(255) |
| IN | **Resource\_Path** – OPTIONAL FILTER: The resource path in which to get/update privileges. | LONGVARCHAR |
| IN | **Resource\_Type** – OPTIONAL FILTER: The resource type in which to get/update privileges. This will only be used when no "Resource\_Path" or a single "Resource\_Path" is provided. It is not used when a list of "Resource\_Path" entries are provided. Example:   * DATA\_SOURCE - a published datasource or physical metadata datasource. * CONTAINER - a folder path, a catalog or schema path. * COLUMN - a column from a table * LINK - a published table or procedure. If it resides in the path /services and points to a TABLE or PROCEDURE then it is a LINK. * TABLE - a view in the /shared path. * PROCEDURE a procedure in the /shared path. | LONGVARCHAR |
| IN | **Name** – OPTIONAL FILTER: The user/group name in which to get/update privileges. | VARCHAR(255) |
| IN | **Name\_Type**– OPTIONAL FILTER: Valid values are USER or GROUP. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to get/update privileges. | VARCHAR(255) |
| IN | **debug** – OPTIONAL: Y=debug on, N=debug off | CHAR(1) |
| OUT | DECLARE PUBLIC TYPE privilegeRowType ROW (  rownum INTEGER,  sheetRownum INTEGER,  datasource VARCHAR(255),  sheetName VARCHAR(255),  privilegeType VARCHAR(255),  Organization VARCHAR(255),  Project VARCHAR(255),  SubProject VARCHAR(255),  "Resource Path" VARCHAR(1024),  "Resource Type" VARCHAR(255),  "Recurse Dependencies" VARCHAR(1),  "Recurse Dependents" VARCHAR(1),  "Recurse Child" VARCHAR(3),  "Revoke All" VARCHAR(1),  Name VARCHAR(255),  "Name Type" VARCHAR(255),  "Domain" VARCHAR(255),  "Env Type" VARCHAR(255),  OrderPrecedence CHAR(1),  R VARCHAR(1),  W VARCHAR(1),  E VARCHAR(1),  S VARCHAR(1),  U VARCHAR(1),  I VARCHAR(1),  D VARCHAR(1),  G VARCHAR(1),  Owner VARCHAR(255),  "Owner Domain" VARCHAR(255),  Comments VARCHAR(1024),  Initialize VARCHAR(255),  isActive CHAR(1) ) | privilegeRowType |

## Privilege Maintenance: Update Privileges

1. updatePrivilegesDriver **–** This script is used to execute the update privileges. This script will access the spreadsheet or database table, batch rows together into a request and submit to DV via the DV admin API. The script runs within the Composite Server. This procedure is used to update privileges by reading the privileges from an excel spreadsheet or database table.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/updatePrivilegesDriver

* 1. The excel sheets include:
     1. Privileges\_shared - Update privileges for /shared
     2. Privileges\_databases - Update privileges for /services/databases
     3. Privileges\_webservices - Update privileges for /services/webservices
     4. COMPOSITE\_PRIVILEGES is a table that contains a type field that indicates which area of composite the row is specifying [shared, databases, webservices].
  2. Pre-requisites to executing the script.
     1. LDAP groups must have been brought into Composite.
     2. The spreadsheet needs to be uploaded to the Composite Server machine.
     3. The spreadsheet data source is updated with the spreadsheet path.
     4. All of the resources being referenced must be present in the Composite server or an exception will be thrown.
     5. The user executing the script must have administrative privileges.

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Batch\_Privileges** – REQUIRED\_FILTER: When the resource path in the spreadsheet changes, a batch is executed. When the resource type in the spreadsheet changes, a batch is executed. When the recursion identifier in the spreadsheet changes, a batch is executed. A group of rows with like recursion may only be batched together otherwise the meaning of the privilege setting is not the same.   * 1=Batch all privileges for the same path, type and recursion setting. * 0=Process each spreadsheet line separately (no batching). | INTEGER |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **Sheet\_Name** – OPTIONAL FILTER: The name of the Excel spreadsheet in which to get/update privileges from: [shared, databases, webservices] | VARCHAR(255) |
| IN | **Resource\_Path** – OPTIONAL FILTER: The resource path in which to get/update privileges. | LONGVARCHAR |
| IN | **Resource\_Type** – OPTIONAL FILTER: The resource type in which to get/update privileges. This will only be used when no "Resource\_Path" or a single "Resource\_Path" is provided. It is not used when a list of "Resource\_Path" entries are provided. Example:   * DATA\_SOURCE - a published datasource or physical metadata datasource. * CONTAINER - a folder path, a catalog or schema path. * COLUMN - a column from a table * LINK - a published table or procedure. If it resides in the path /services and points to a TABLE or PROCEDURE then it is a LINK. * TABLE - a view in the /shared path. * PROCEDURE a procedure in the /shared path. | LONGVARCHAR |
| IN | **Name** – OPTIONAL FILTER: The user/group name in which to get/update privileges. | VARCHAR(255) |
| IN | **Name\_Type**– OPTIONAL FILTER: Valid values are USER or GROUP. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to get/update privileges. | VARCHAR(255) |
| IN | **preview** – N or null(default)=Do not preview. Execute the privileges. Y=preview what will get set but don't actually execute the operation. | CHAR(1) |
| IN | **forceDomainAsComposite** – OPTIONAL PARAM: Y=true, N=false. For example, the LDAP group DV\_Demo\_Dev is automatically converted to the composite domain group dv\_demo\_dev.   * Provides a way to force a conversion of the group from an LDAP domain name like "ldap" to the "composite" domain. * This is useful when setting up the privileges for testing with composite groups that mirror the LDAP groups. This way the spreadsheet does not have to be modified. | CHAR(1) |
| IN | **inDebug1** – N/Null=No debug, Y (default)=Debug for this procedure - basic messages. | CHAR(1) |
| IN | **inDebug1Console** – N/Null (default)=No debug, Y=Debug console. Determines whether to print to the Studio console window or not. | CHAR(1) |
| IN | **inDebug1CISLog** – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the DV log or not. | CHAR(1) |
| IN | **inDebug1ReadOnly** – N/Null (default)=No debug, Y=Debug for READ only messages. | CHAR(1) |
| IN | **inDebug1RevokeAll**  – N/Null (default)=No debug, Y=Debug for Revoke All messages. | CHAR(1) |
| IN | **inDebug2** – N/Null (default)=No debug, Y=Debug for 2nd and 3rd tier procedures (deep debug). | CHAR(1) |
| IN | **inBypassErrors** – Bypass errors. Throw exception when paths not found.   * N/Null (default) Do not bypass errors. * Y=bypass resource not found errors but report them. | CHAR(1) |
| OUT | **rowsProcessed** – number of rows processed from the spreadsheet | INTEGER |
| OUT | **logOutput** – The output log that gets printed to the command line console is also output. | LONGVARCHAR |

## Privilege Maintenance: Validate Privileges

1. validatePrivilegesDriver **–** This script is used to validate privileges. This procedure is used to validate privileges by reading the privileges from an excel spreadsheet and validate the resource path exists and the name (USER or GROUP) exists in the specified domain and the privilege is set according to the spreadsheet or database setting for the filter applied.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/validatePrivilegesDriver

* 1. Read the privileges from the following sheets:
     1. Privileges\_shared - /shared privileges
     2. Privileges\_databases - /services/databases privileges
     3. Privileges\_webservices - /services/webservices privileges

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **Sheet\_Name** – OPTIONAL FILTER: The name of the Excel spreadsheet in which to get/update privileges from: [shared, databases, webservices] | VARCHAR(255) |
| IN | **Resource\_Path** – OPTIONAL FILTER: The resource path in which to get/update privileges. | LONGVARCHAR |
| IN | **Resource\_Type** – OPTIONAL FILTER: The resource type in which to get/update privileges. This will only be used when no "Resource\_Path" or a single "Resource\_Path" is provided. It is not used when a list of "Resource\_Path" entries are provided. Example:   * DATA\_SOURCE - a published datasource or physical metadata datasource. * CONTAINER - a folder path, a catalog or schema path. * COLUMN - a column from a table * LINK - a published table or procedure. If it resides in the path /services and points to a TABLE or PROCEDURE then it is a LINK. * TABLE - a view in the /shared path. * PROCEDURE a procedure in the /shared path. | LONGVARCHAR |
| IN | **Name** – OPTIONAL FILTER: The user/group name in which to get/update privileges. | VARCHAR(255) |
| IN | **Name\_Type**– OPTIONAL FILTER: Valid values are USER or GROUP. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to get/update privileges. | VARCHAR(255) |
| IN | **forceDomainAsComposite** – OPTIONAL PARAM: Y=true, N=false. For example, the LDAP group DV\_Demo\_Dev is automatically converted to the composite domain group dv\_demo\_dev.   * Provides a way to force a conversion of the group from an LDAP domain name like "ldap" to the "composite" domain. * This is useful when setting up the privileges for testing with composite groups that mirror the LDAP groups. This way the spreadsheet does not have to be modified. | CHAR(1) |
| IN | **inDebug1** – N/Null=No debug, Y (default)=Debug for this procedure - basic messages. | CHAR(1) |
| IN | **inDebug1Console** – N/Null (default)=No debug, Y=Debug console. Determines whether to print to the Studio console window or not. | CHAR(1) |
| IN | **inDebug1CISLog** – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the DV log or not. | CHAR(1) |
| IN | **inDebug1ReadOnly** – N/Null (default)=No debug, Y=Debug for READ only messages. | CHAR(1) |
| IN | **inDebug2** – N/Null (default)=No debug, Y=Debug for 2nd and 3rd tier procedures (deep debug). | CHAR(1) |
| IN | **inBypassErrors** – Bypass errors. Throw exception when paths not found.   1. N/Null (default) Do not bypass errors. 2. Y=bypass resource not found errors but report them. | CHAR(1) |
| OUT | **validPrivileges** – true=if all privileges are validated. false=if one or more privileges are invalid. | VARCHAR |
| OUT | **rowsProcessed** – number of rows processed from the spreadsheet | INTEGER |
| OUT | **invalidPrivilegeList** – Pipe separated list of invalid pivileges. | LONGVARCHAR |
| OUT | **invalidResourceList** – Pipe separated list of invalid resource paths that do not exist. | LONGVARCHAR |
| OUT | **invalidNameList** – Pipe separated list of name, nameType, and domain combination that does not exist. | LONGVARCHAR |
| OUT | **logOutput** – The output log that gets printed to the command line console is also output. | LONGVARCHAR |

## Group Maintenance: Get Groups

1. getGroups **–** Construct a SQL statement based on filters passed in and retrieve the set of groups that match the criteria.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/getGroups

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **Group\_Name** – OPTIONAL FILTER: The group name in which to get/update groups. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to get/update groups. | VARCHAR(255) |
| IN | **debug** – OPTIONAL: Y=debug on, N=debug off | CHAR(1) |
| OUT | DECLARE PUBLIC TYPE groupsRowType ROW (  rownum INTEGER,  sheetRownum INTEGER,  datasource VARCHAR(255),  sheetName VARCHAR(255),  Organization VARCHAR(255),  Project VARCHAR(255),  SubProject VARCHAR(255),  GroupName VARCHAR(255),  Description VARCHAR(1024),  "Domain" VARCHAR(255),  "Env Type" VARCHAR(255),  Access VARCHAR(1),  Unlock VARCHAR(1),  Config\_R VARCHAR(1),  Config\_M VARCHAR(1),  Resources\_R VARCHAR(1),  Resources\_M VARCHAR(1),  Status\_R VARCHAR(1),  Status\_M VARCHAR(1),  Users\_R VARCHAR(1),  Users\_M VARCHAR(1),  IsActive VARCHAR(1) ) | groupsRowType |

## Group Maintenance: Update Groups

1. updateGroupsDriver **–** This script is used to create or update groups. Reads the Composite Privileges spreadsheet and gets the list of groups from the sheet: Group\_List

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/updateGroupsDriver

* 1. Limitations
     1. If the domain is "composite" and the group does not exist then the group is created.
     2. If the domain is "composite" and the group does exist then the group is updated.
     3. If the domain is LDAP and the group does not exist then the group is NOT created and not updated.
     4. If the domain is LDAP and the group does exist then the group is updated.

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **Group\_Name**–OPTIONAL FILTER*:* The group name for which to get/update groups. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to get/update groups. | VARCHAR(255) |
| IN | **preview** – N or null(default)=Do not preview. Execute the privileges. Y=preview what will get set but don't actually execute the operation. | CHAR(1) |
| IN | **forceDomainAsComposite** – OPTIONAL PARAM: Y=true, N=false. For example, the LDAP group DV\_Demo\_Dev is automatically converted to the composite domain group dv\_demo\_dev.   * Provides a way to force a conversion of the group from an LDAP domain name like "ldap" to the "composite" domain. * This is useful when setting up the privileges for testing with composite groups that mirror the LDAP groups. This way the spreadsheet does not have to be modified. | CHAR(1) |
| IN | **inDebug1** – N/Null=No debug, Y (default)=Debug for this procedure - basic messages. | CHAR(1) |
| IN | **inDebug1Console** – N/Null (default)=No debug, Y=Debug console. Determines whether to print to the Studio console window or not. | CHAR(1) |
| IN | **inDebug1CISLog** – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the DV log or not. | CHAR(1) |
| IN | **inDebug1ReadOnly** – N/Null (default)=No debug, Y=Debug for READ only messages. | CHAR(1) |
| OUT | **rowsProcessed** – number of rows processed from the spreadsheet | INTEGER |
| OUT | **logOutput** – The output log that gets printed to the command line console is also output. | LONGVARCHAR |

## Group Maintenance: Delete Groups

1. deleteGroupsDriver **–** This script is used to delete groups from DV. Reads the Composite Privileges spreadsheet and gets the list of groups from the sheet: Group\_List

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/deleteGroupsDriver

* 1. Limitations
     1. If the group does not exist then the group is NOT deleted.
     2. If the group does exist then the group is deleted no matter what domain the group is in.

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **Group\_Name**–OPTIONAL FILTER*:* The group name for which to delete groups. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to delete groups. | VARCHAR(255) |
| IN | **preview** – N or null(default)=Do not preview. Execute the privileges. Y=preview what will get set but don't actually execute the operation. | CHAR(1) |
| IN | **forceDomainAsComposite** – OPTIONAL PARAM: Y=true, N=false. For example, the LDAP group DV\_Demo\_Dev is automatically converted to the composite domain group dv\_demo\_dev.   * Provides a way to force a conversion of the group from an LDAP domain name like "ldap" to the "composite" domain. * This is useful when setting up the privileges for testing with composite groups that mirror the LDAP groups. This way the spreadsheet does not have to be modified. | CHAR(1) |
| IN | **inDebug1** – N/Null=No debug, Y (default)=Debug for this procedure - basic messages. | CHAR(1) |
| IN | **inDebug1Console** – N/Null (default)=No debug, Y=Debug console. Determines whether to print to the Studio console window or not. | CHAR(1) |
| IN | **inDebug1CISLog** – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the DV log or not. | CHAR(1) |
| OUT | **rowsProcessed** – number of rows processed from the spreadsheet | INTEGER |
| OUT | **logOutput** – The output log that gets printed to the command line console is also output. | LONGVARCHAR |

## Group Maintenance: Validate Groups

1. validateGroupsDriver **–** This script is used to validate groups. Reads the Composite Privileges spreadsheet and validates whether the user exists or not from the sheet: Group\_List

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/validateGroupsDriver

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **Group\_Name**–OPTIONAL FILTER*:* The group name for which to validate groups. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to validate groups. | VARCHAR(255) |
| IN | **forceDomainAsComposite** – OPTIONAL PARAM: Y=true, N=false. For example, the LDAP group DV\_Demo\_Dev is automatically converted to the composite domain group dv\_demo\_dev.   * Provides a way to force a conversion of the group from an LDAP domain name like "ldap" to the "composite" domain. * This is useful when setting up the privileges for testing with composite groups that mirror the LDAP groups. This way the spreadsheet does not have to be modified. | CHAR(1) |
| IN | **inDebug1** – N/Null=No debug, Y (default)=Debug for this procedure - basic messages. | CHAR(1) |
| IN | **inDebug1Console** – N/Null (default)=No debug, Y=Debug console. Determines whether to print to the Studio console window or not. | CHAR(1) |
| IN | **inDebug1CISLog** – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the DV log or not. | CHAR(1) |
| IN | **inDebug2** – N/Null (default)=No debug, Y=Debug for 2nd and 3rd tier procedures (deep debug). | CHAR(1) |
| OUT | **validGroups** – true=if all groups are present (valid). false=if one or more groups are not present (invalid). | VARCHAR |
| OUT | **rowsProcessed** – number of rows processed from the spreadsheet | INTEGER |
| OUT | **logOutput** – The output log that gets printed to the command line console is also output. | LONGVARCHAR |
| OUT | **groupResult** – Result with pipe dlimited rows  Format of output for a single row within the pipe:  **Row=**row number from spreadsheet  **SheetRow=**the number of row within the sheet  **Sheet=**The name of the sheet from the spreadsheet “Group\_List”  **groupName=**the name of the group  **domainName=**the name of the domain  **exists=**YES or NO indicating whether the group exists or not.  **studio\_rights=**YES or NO indicating if the studio rights are valid if the row exists  diff\_rights=Provides a list of studio rights where each right contains [=,+,-] in front. The “=” means that the studio right from the source is the same as the actual set in DV. The “+” means that the studio right is set in DV but not supposed to be set as per the spreadsheet. The “-“ means that the studio right is not set in DV but it is supposed to be set as per the spreadsheet. Example: =ACCESS\_TOOLS +UNLOCK\_RESOURCE +READ\_ALL\_CONFIG +MODIFY\_ALL\_CONFIG -READ\_ALL\_RESOURCES  **|=** end of the row | LONGVARCHAR |

## Group Maintenance: Get Users

1. getUsers **–** Construct a SQL statement based on filters passed in and retrieve the set of users that match the criteria.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/getUsers

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **User\_Name** – OPTIONAL FILTER: The user name in which to get/update users. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to get/update users. | VARCHAR(255) |
| IN | **debug** – OPTIONAL: Y=debug on, N=debug off | CHAR(1) |
| OUT | DECLARE PUBLIC TYPE userRowType ROW (  rownum INTEGER,  sheetRownum INTEGER,  datasource VARCHAR(255),  sheetName VARCHAR(255),  Organization VARCHAR(255),  Project VARCHAR(255),  SubProject VARCHAR(255),  UserName VARCHAR(255),  Description VARCHAR(1024),  "Domain" VARCHAR(255),  "Env Type" VARCHAR(255),  Password VARCHAR(255),  Force CHAR(1),  GroupList VARCHAR(255),  UserName\_lowercase VARCHAR(255),  IsActive VARCHAR(1) ) | userRowType |

## User Maintenance: Update Users

These scripts are used for rebinding the generation scripts from one folder to another.

1. updateUsersDriver–This script is used to create or update composite users only. It does not create or update LDAP users. Reads the Composite Privileges spreadsheet and gets the list of users from the sheet: User\_List

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/updateUsersDriver

* 1. Limitations:
     1. If the domain is "composite" and the user does not exist then the user is created.
     2. If the domain is "composite" and the user does exist then the user is updated.
     3. If the domain is LDAP no action is taken at all as users are not maintained in Composite.

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **User\_Name**–OPTIONAL FILTER*:* The user name for which to create/update users. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to get/update users. | VARCHAR(255) |
| IN | **preview** – N or null(default)=Do not preview. Execute the privileges. Y=preview what will get set but don't actually execute the operation. | CHAR(1) |
| IN | **inDebug1** – N/Null=No debug, Y (default)=Debug for this procedure - basic messages. | CHAR(1) |
| IN | **inDebug1Console** – N/Null (default)=No debug, Y=Debug console. Determines whether to print to the Studio console window or not. | CHAR(1) |
| IN | **inDebug1CISLog** – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the DV log or not. | CHAR(1) |
| OUT | **rowsProcessed** – number of rows processed from the spreadsheet | INTEGER |
| OUT | **logOutput** – The output log that gets printed to the command line console is also output. | LONGVARCHAR |

## User Maintenance: Delete Users

1. deleteUsersDriver **–** This script is used to delete composite users only. It does not create or update LDAP users. Reads the Composite Privileges spreadsheet and gets the list of users from the sheet: User\_List.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/deleteUsersDriver

* 1. Limitations:
     1. If the domain is "composite" and the user does not exist then the user is NOT deleted.
     2. If the domain is "composite" and the user does exist then the user is deleted.
     3. If the domain is LDAP and the user does not exist then the user is NOT deleted.
     4. If the domain is LDAP and the user does exist then the user is NOT deleted.

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL - Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE - SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD - SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **User\_Name**–OPTIONAL FILTER*:* The user name for which to delete users. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to delete users. | VARCHAR(255) |
| IN | **preview** – N or null(default)=Do not preview. Execute the privileges. Y=preview what will get set but don't actually execute the operation. | CHAR(1) |
| IN | **inDebug1** – N/Null=No debug, Y (default)=Debug for this procedure - basic messages. | CHAR(1) |
| IN | **inDebug1Console** – N/Null (default)=No debug, Y=Debug console. Determines whether to print to the Studio console window or not. | CHAR(1) |
| IN | **inDebug1CISLog** – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the DV log or not. | CHAR(1) |
| OUT | **rowsProcessed** – number of rows processed from the spreadsheet | INTEGER |
| OUT | **logOutput** – The output log that gets printed to the command line console is also output. | LONGVARCHAR |

## User Maintenance: Validate Users

1. validateUsersDrivers **–** This script is used to validate users. Reads the Composite Privileges spreadsheet and validates whether the group exists or not using the sheet: User\_List.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/validateUsersDriver

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **Datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **Environment\_Name** – REQUIRED FILTER: The Composite server environment in which to get/update privileges for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **Organization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **Project** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this “Env Type”. This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| IN | **SubProject**–OPTIONAL FILTER*:* The name of the sub-project that is hosted on DV for this “Env Type”. This provides a filter for only applying privileges for the given sub-project. If left blank, all rows for the project will be updated. | VARCHAR(255) |
| IN | **User\_Name**–OPTIONAL FILTER*:* The user name for which to validate users. | VARCHAR(255) |
| IN | **Domain\_Name** – OPTIONAL FILTER: The domain name in which to validate users. | VARCHAR(255) |
| IN | **inDebug1** – N/Null=No debug, Y (default)=Debug for this procedure – basic messages. | CHAR(1) |
| IN | **inDebug1Console** – N/Null (default)=No debug, Y=Debug console. Determines whether to print to the Studio console window or not. | CHAR(1) |
| IN | **inDebug1CISLog** – N/Null (default)=No debug, Y=Debug log. Determines whether to print to the DV log or not. | CHAR(1) |
| IN | **inDebug2** – N/Null (default)=No debug, Y=Debug for 2nd and 3rd tier procedures (deep debug). | CHAR(1) |
| OUT | **validUsers** – true=if all users are present (valid). False=if one or more users are not present (invalid). | VARCHAR |
| OUT | **rowsProcessed** – number of rows processed from the spreadsheet | INTEGER |
| OUT | **logOutput** – The output log that gets printed to the command line console is also output. | LONGVARCHAR |
| OUT | **userResult** – Result with pipe delimited rows | LONGVARCHAR |

## Database Maintenance: Load Database Privileges

1. loadPrivileges\_EXCEL **–** This procedure is the generic interface for inserting and updating privileges from the generic Resource\_Privileges\_LOAD\_DB.xlsx spreadsheet. The location of the resource in DV is as follows: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/Privileges\_DS\_EXCEL/Resource\_Privileges\_LOAD\_DB.xlsx.

It does not delete privilege rows. Any deletion of privileges should be done through “modify\_06\_COMPOSITE\_PRIVILEGES”. In order to update a row, it uses the following columns to detect the existence of a row: ISACTIVE=Y, ORGANIZATION\_NAME, PROJECT\_NAME, SUBPROJECT\_NAME, RESOURCE\_PATH, RESOURCE\_TYPE, DOMAIN, NAME\_TYPE, NAME, and ENVIRONMENT\_NAME. If no rows are detected then an insert is performed.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/loadPrivileges\_EXCEL

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **inPerformInsertUpdate** – Y or N. Y=yes perform insert or update. N=no do not perform insert or update but display the same messages to see what will be executed. | CHAR(1) |
| OUT | **rowsProcessed** – Number of rows processed in total. | INTEGER |
| OUT | **rowsInserted** – Number of rows inserted into the privilege database. | INTEGER |
| OUT | **rowsUpdated** – Number of rows updated into the privilege database. | INTEGER |
| OUT | **rowsDeleted** – Number of rows deleted into the privilege database. | INTEGER |
| OUT | **rowsInactive** – Number of rows set to inactive into the privilege database. | INTEGER |
| OUT | **duplicateRows** – number of duplicate rows. | INTEGER |
| OUT | **duplicateRowMsg** – message for duplicate rows. | LONGVARCHAR |
| OUT | **message** – a general message | LONGVARCHAR |

## Database Maintenance: Load Database Groups

1. loadGroups\_EXCEL **–** This procedure is the generic interface for inserting and updating groups from the generic Resource\_Privileges\_LOAD\_DB.xlsx spreadsheet. The location of the resource in DV is as follows: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/Privileges\_DS\_EXCEL/Resource\_Privileges\_LOAD\_DB.xlsx.

It does not delete group rows. Any deletion of groups should be done through “modify\_07\_COMPOSITE\_STUDIO\_RIGHTS”. In order to update a row, it uses the following columns to detect the existence of a row: ISACTIVE=Y, ORGANIZATION\_NAME, PROJECT\_NAME, GROUP\_NAME, DOMAIN, ENVIRONMENT\_NAME. If no rows are detected then an insert is performed.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/loadGroups\_EXCEL

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **inPerformInsertUpdate** – Y or N. Y=yes perform insert or update. N=no do not perform insert or update but display the same messages to see what will be executed. | CHAR(1) |
| IN | **useDefaultConfig** – Y or N. Y=use the default config values for the following thus overriding the spreadsheet values.  READ\_ALL\_CONFIG=null - this value unset  MODIFY\_ALL\_CONFIG=null - this value unset  READ\_ALL\_RESOURCES=null - this value unset  MODIFY\_ALL\_RESOURCES=null- this value unset | CHAR(1) |
| OUT | **rowsProcessed** – Number of rows processed in total. | INTEGER |
| OUT | **rowsInserted** – Number of rows inserted into the privilege database. | INTEGER |
| OUT | **rowsUpdated** – Number of rows updated into the privilege database. | INTEGER |
| OUT | **rowsDeleted** – Number of rows deleted into the privilege database. | INTEGER |
| OUT | **rowsInactive** – Number of rows set to inactive into the privilege database. | INTEGER |
| OUT | **duplicateRows** – number of duplicate rows. | INTEGER |
| OUT | **duplicateRowMsg** – message for duplicate rows. | LONGVARCHAR |
| OUT | **message** – a general message | LONGVARCHAR |

## Database Maintenance: Load Database Users

1. loadUsers\_EXCEL **–** This procedure is the generic interface for inserting and updating users from the generic Resource\_Privileges\_LOAD\_DB.xlsx spreadsheet. The location of the resource in DV is as follows: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/Metadata/Privileges\_DS\_EXCEL/Resource\_Privileges\_LOAD\_DB.xlsx.

It does not delete user rows. Any deletion of groups should be done through “modify\_08\_COMPOSITE\_USER”. In order to update a row, it uses the following columns to detect the existence of a row: ISACTIVE=Y, ORGANIZATION\_NAME, PROJECT\_NAME, SUBPROJECT\_NAME, USER\_NAME, DOMAIN, ENVIRONMENT\_NAME. If no rows are detected then an insert is performed.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/loadUsers\_EXCEL

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **inPerformInsertUpdate** – Y or N. Y=yes perform insert or update. N=no do not perform insert or update but display the same messages to see what will be executed. | CHAR(1) |
| OUT | **rowsProcessed** – Number of rows processed in total. | INTEGER |
| OUT | **rowsInserted** – Number of rows inserted into the privilege database. | INTEGER |
| OUT | **rowsUpdated** – Number of rows updated into the privilege database. | INTEGER |
| OUT | **rowsDeleted** – Number of rows deleted into the privilege database. | INTEGER |
| OUT | **rowsInactive** – Number of rows set to inactive into the privilege database. | INTEGER |
| OUT | **duplicateRows** – number of duplicate rows. | INTEGER |
| OUT | **duplicateRowMsg** – message for duplicate rows. | LONGVARCHAR |
| OUT | **message** – a general message | LONGVARCHAR |

## Database Maintenance: Perform Supporting Table Maintenance

1. performSupportingTableMaintenance **–** This procedure is used to perform table maintenance on the supporting tables in the database.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/performSupportingTableMaintenance

Table Name Require Fields

ORGANIZATION\_TYPE: ORGANIZATION\_NAME

PROJECT\_TYPE: ORGANIZATION\_NAME

PROJECT\_NAME

SUBPROJECT\_TYPE: SUBPROJECT\_NAME

PROJECT\_NAME

ORGANIZATION\_NAME

DOMAIN\_TYPE: DOMAIN\_NAME

ENVIRONMENT\_TYPE: ENVIRONMENT\_NAME

Rules:

1. performInsert:

When performInsert=Y, perform the insert of the new row for the given table.

When performInsert=N, do not perform the insert of the new row for the given table but display what will happen.

2. organization

Used for tables ORGANIZATION\_TYPE or PROJECT\_TYPE

If not null, then test ORGANIZATION\_TYPE.ORGANIZATION\_NAME for existence.

If not exist then insert.

3. projectName

Used for table PROJECT\_TYPE.ORGANIZATION\_NAME and PROJECT\_TYPE.PROJECT\_NAME

If projectName and organization is not null then test PROJECT\_TYPE.ORGANIZATION\_NAME and PROJECT\_TYPE.PROJECT\_NAME for existence.

If not exist then insert.

4. subprojectList

Used for table SUBPROJECT\_TYPE.SUBPROJECT\_NAME and PROJECT\_NAME and ORGANIZATION\_NAME

This may be a comma separated list of subprojects.

For each subproject and project, if they are not null then test PROJECT\_TYPE.SUBPROJECT\_NAME and PROJECT\_NAME and ORGANIZATION\_NAME for existence.

If not exist then insert.

5. domainNameList

Used for tables DOMAIN\_TYPE

This may be a comma separated list of domain names.

If not null, then test DOMAIN\_TYPE.DOMAIN\_NAME for existence.

If not exist then insert.

6. environmentNameList

Used for tables ENVIRONMENT\_TYPE

This may be a comma separated list of environent names.

If not null, then test ENVIRONMENT\_TYPE.ENVIRONMENT\_NAME for existence.

If not exist then insert.

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **performInsert** – Y or N. Y=yes perform insert or update. N=no do not perform insert or update but display the same messages to see what will be executed. | CHAR(1) |
| IN | **organization** – leave null unless you want to insert into ORGANIZATION\_TYPE and/or PROJECT\_TYPE. Examples (1 of): ORG1, ORG2, ORG3. | VARCHAR |
| IN | **projectName** – leave null unless you want to insert into PROJECT\_TYPE and/or SUBPROJECT\_TYPE. Examples (1 of): PROJECT1, PROJECT2. | VARCHAR |
| IN | **subprojectList** – leave null unless you want to insert into SUBPROJECT\_TYPE. This may be a comma separated list of subprojects. Examples (1 of): SB1, SB2, SB3. | LONGVARCHAR |
| IN | **domainName List**– leave null unless you want to insert into DOMAIN\_TYPE. This may be a comma separated list of domain names. Examples (1 of): ldap, composite, dynamic | LONGVARCHAR |
| IN | **environemntNameList** – leave null unless you want to insert into ENVIRONMENT\_TYPE. This may be a comma separated list of environment names. Examples (1 of): DEV, UAT, PROD | LONGVARCHAR |
| OUT | **message** – a general message | LONGVARCHAR |

## Database Maintenance: Fix Privilege Table

1. fix\_PROD\_ORA\_COMPOSITE\_PRIVILEGES **–** This procedure is used to perform table maintenance on the COMPOSITE\_PRIVILEGES Oracle PROD table to fix any issues.
   1. fix\_PROD\_SS\_COMPOSITE\_PRIVILEGES **–** This procedure is used to perform table maintenance on the COMPOSITE\_PRIVILEGES SQL Server PROD table to fix any issues.
   2. The following columns should not contain any blanks, spaces, empty character or any characters other than [X,Y,N]. This procedure updates the COMPOSITE\_PRIVILEGES table and sets the column to NULL if it does not X, Y and N. The column being updated include: INITIALIZE, READ\_ACCESS, WRITE\_ACCESS, EXECUTE\_ACCESS, SELECT\_ACCESS, UPDATE\_ACCESS, INSERT\_ACCESS, DELETE\_ACCESS, GRANT\_ACCESS.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/TableMaintenance/fix\_...

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **inOrganization** – RECOMMENDED FILTER: The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **inProject** – RECOMMENDED FILTER: The name of the project that is hosted on DV for this "Env Type". This provides a filter for only applying privileges for the given project. If left blank, all projects will be updated. | VARCHAR(255) |
| OUT | **countINITIALIZE** – number of rows modified. | INTEGER |
| OUT | **countREAD\_ACCESS** – number of rows modified. | INTEGER |
| OUT | **countWRITE\_ACCESS** – number of rows modified. | INTEGER |
| OUT | **countEXECUTE\_ACCESS** – number of rows modified. | INTEGER |
| OUT | **countSELECT\_ACCESS** – number of rows modified. | INTEGER |
| OUT | **countUPDATE\_ACCESS** – number of rows modified. | INTEGER |
| OUT | **countINSERT\_ACCESS** – number of rows modified. | INTEGER |
| OUT | **countDELETE\_ACCESS** – number of rows modified. | INTEGER |
| OUT | **countGRANT\_ACCESS** – number of rows modified. | INTEGER |
| OUT | **countOWNER\_NAME\_DEV** – number of rows modified. | INTEGER |
| OUT | **countOWNER\_NAME\_UAT** – number of rows modified. | INTEGER |
| OUT | **countOWNER\_NAME\_PROD** – number of rows modified. | INTEGER |
| OUT | **countOWNER\_DOMAIN\_DEV** – number of rows modified. | INTEGER |
| OUT | **countOWNER\_DOMAIN\_UAT** – number of rows modified. | INTEGER |
| OUT | **countOWNER\_DOMAIN\_PROD** – number of rows modified. | INTEGER |

## Database Maintenance: Modify Organization Type

1. modify\_01\_ORGANIZATION\_TYPE **–** This procedure is used to insert/update/delete/inactivate rows in the ORGANIZATION\_TYPE table.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/TableMaintenance/ modify\_01\_ORGANIZATION\_TYPE

Examples:

ORG1

ORG2

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **setInactive** – Set the ISACTIVE flag to N. Y=if the record exists set it inactive. Overrides performDelete. N/null=do not perform this operation. | VARCHAR(1) |
| IN | **performDelete** – Delete the row if it exists. Y=perform a delete using filters supplied. N/null=do not perform delete but perform insert or update instead. | VARCHAR(1) |
| IN | **inORGANIZATION\_NAME** – The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| OUT | **message** – A general message | LONGVARCHAR |

## Database Maintenance: Modify Domain Type

1. modify\_02\_DOMAIN\_TYPE **–** This procedure is used to insert/update/delete/inactivate rows in the DOMAIN\_TYPE table. The two composite domains are "composite" and "dynamic". Additional domain names for LDAP will be whatever the administrator chooses such as "ldap".

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/TableMaintenance/ modify\_02\_DOMAIN\_TYPE

Examples:

ldap

composite

dynamic

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **setInactive** – Set the ISACTIVE flag to N. Y=if the record exists set it inactive. Overrides performDelete. N/null=do not perform this operation. | VARCHAR(1) |
| IN | **performDelete** – Delete the row if it exists. Y=perform a delete using filters supplied. N/null=do not perform delete but perform insert or update instead. | VARCHAR(1) |
| IN | **inDOMAIN\_NAME** – The domain name to insert/update/delete/inactivate. If both setInactive and performDelete are null or N then perform the insert. If the row does not exist then insert. If the row already exists then do nothing. | VARCHAR(255) |
| OUT | **message** – A general message | LONGVARCHAR |

## Database Maintenance: Modify Environment Type

1. modify\_03\_ENVIRONMENT\_TYPE **–** This procedure is used to insert/update/delete/inactivate rows in the ENVIRONMENT\_TYPE table. The environment will match the LDAP server environment. In some organizations, they will have an environment for each deployment area such as DEV, UAT and PROD. Other organizations may only have a single LDAP environment such as PROD in which all DV instances are connected to. The name is an abstract name but descriptive of the LDAP instance that the DV domain is connecting to.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/TableMaintenance/ modify\_03\_ENVIRONMENT\_TYPE

Examples:

DEV

UAT

PROD

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **setInactive** – Set the ISACTIVE flag to N. Y=if the record exists set it inactive. Overrides performDelete. N/null=do not perform this operation. | VARCHAR(1) |
| IN | **performDelete** – Delete the row if it exists. Y=perform a delete using filters supplied. N/null=do not perform delete but perform insert or update instead. | VARCHAR(1) |
| IN | **inENVIRONMENT\_NAME** – The environment name to insert/update/delete/inactivate. If both setInactive and performDelete are null or N then perform the insert. If the row does not exist then insert. If the row already exists then do nothing. | VARCHAR(255) |
| OUT | **message** – A general message | LONGVARCHAR |

## Database Maintenance: Modify Project Type

1. modify\_04\_PROJECT\_TYPE **–** This procedure is used to insert/update/delete/inactivate rows in the PROJECT\_TYPE table. This procedure performs a cascade modify for the ORGANIZATION\_TYPE table also. However, it does not perform the cascade inactivate. That must be done explicitly on each table.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/TableMaintenance/ modify\_04\_PROJECT\_TYPE

Examples:

PROJECT\_NAME ORGANIZATION\_NAME

PROJECT1 ORG1

PROJECT2 ORG2

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **setInactive** – Set the ISACTIVE flag to N. Y=if the record exists set it inactive. Overrides performDelete. N/null=do not perform this operation. | VARCHAR(1) |
| IN | **performDelete** – Delete the row if it exists. Y=perform a delete using filters supplied. N/null=do not perform delete but perform insert or update instead. | VARCHAR(1) |
| IN | **inPROJECT** – The Composite server environment in which to update studio rights for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **inORGANIZATION\_NAME** – The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| OUT | **message** – A general message | LONGVARCHAR |

## Database Maintenance: Modify SubProject Type

1. modify\_05\_SUBPROJECT\_TYPE **–** This procedure is used to insert/update/delete/inactivate rows in the SUBPROJECT\_TYPE table. This procedure performs a cascade modify for the PROJECT\_TYPE table also. This procedure performs a cascade modify for the ORGANIZATION\_TYPE table also. However, it does not perform the cascade inactivate. That must be done explicitly on each table.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/TableMaintenance/ modify\_05\_SUBPROJECT\_TYPE

Examples:

SUBPROJECT\_NAME PROJECT\_NAME

SB1 PROJECT1

SB2 PROJECT2

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **setInactive** – Set the ISACTIVE flag to N. Y=if the record exists set it inactive. Overrides performDelete. N/null=do not perform this operation. | VARCHAR(1) |
| IN | **performDelete** – Delete the row if it exists. Y=perform a delete using filters supplied. N/null=do not perform delete but perform insert or update instead. | VARCHAR(1) |
| IN | **inSUBPROJECT\_NAME** – The subproject name to insert/update/delete/inactivate. If both setInactive and performDelete are null or N then perform the insert. If the row does not exist then insert. If the row already exists then do nothing. | VARCHAR(255) |
| IN | **inPROJECT** – The Composite server environment in which to update studio rights for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **inORGANIZATION\_NAME** – The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| OUT | **message** – A general message | LONGVARCHAR |

## Database Maintenance: Modify Composite Privileges

1. modify\_06\_COMPOSITE\_PRIVILEGES **–** This procedure is used to insert/update/delete/inactivate rows in the COMPOSITE\_PRIVILEGES table. These are rows that come from the Privileges\_shared, Privileges\_databases, and Privileges\_webservices sheets in the Resource\_Privileges\_LOAD\_DB.xlsx spreadsheet.

For deleting records the following fields are used to create the where clause: PRIVIELGE\_ID - if present then it only uses this to delete that 1 record.

The following list of fields are used for constructing a dynamic where clause. If a field contains a null then it is not used for the where clause. Therefore it provides a lot of flexibility for deleting as much or little as you wish. inENVIRONMENT\_NAME, inORGANIZATION\_NAME, inPROJECT\_NAME, inSUBPROJECT\_NAME, inRESOURCE\_PATH, inNAME, inNAME\_TYPE, inDOMAIN\_NAME

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/TableMaintenance/ modify\_06\_COMPOSITE\_PRIVILEGES

Example:

PRIVILEGE\_ID: 2

ORGANIZATION\_NAME: ORG1

PRIVILEGE\_TYPE: SHARED

PROJECT\_NAME: Common

SUBPROJECT\_NAME: [NULL]

RESOURCE\_PATH: /shared/examples

RESOURCE\_TYPE: CONTAINER

RECURSE\_DEPENDENCIES: [NULL]

RECURSE\_DEPENDENTS: [NULL]

RECURSE\_CHILD: YM

REVOKE\_ALL: N

NAME: group1

NAME\_TYPE: GROUP

DOMAIN\_NAME: composite

ENVIRONMENT\_NAME: DEV

READ\_ACCESS: X

WRITE\_ACCESS: [NULL]

EXECUTE\_ACCESS: X

SELECT\_ACCESS: X

UPDATE\_ACCESS: [NULL]

INSERT\_ACCESS: [NULL]

DELETE\_ACCESS: [NULL]

GRANT\_ACCESS: [NULL]

INITIALIZE: X

OWNER\_NAME: [NULL]

OWNER\_DOMAIN: [NULL]

COMMENTS: initialize

CREATEDBY: CMPPRV

CREATEDDATE: 2017-05-17 15:31:13

UPDATEDBY: CMPPRV

UPDATEDDATE: 2017-05-17 15:31:13

ISACTIVE: Y

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **performInsertUpdate** – Y or N. Y=yes perform insert or update. N=no do not perform insert or update but display the same messages to see what will be executed. | VARCHAR(1) |
| IN | **setInactive** – Set the ISACTIVE flag to N. Y=if the record exists set it inactive. Overrides performDelete. N/null=do not perform this operation. | VARCHAR(1) |
| IN | **performDelete** – Delete the row if it exists. Y=perform a delete using filters supplied. N/null=do not perform delete but perform insert or update instead. | VARCHAR(1) |
| IN | **inPRIVILEGE\_ID** – This is used only when setting a row inactive or deleting a specific row by its generated PRIVILEGE\_ID. | DECIMAL(38,0) |
| IN | **inORGANIZATION\_NAME** – The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **inPROJECT\_NAME** – The project name to insert/update/delete/inactivate. If both setInactive and performDelete are null or N then perform the insert. If the row does not exist then insert. If the row already exists then do nothing. | VARCHAR(255) |
| IN | **inSUBPROJECT\_NAME** – The subproject name to insert/update/delete/inactivate. | VARCHAR(255) |
| IN | **inRESOURCE\_PATH** – The DV resource path. | VARCHAR(4000) |
| IN | **inRESOURCE\_TYPE** – The DV resource type such as DATA\_SOURCE, CONTAINER, TABLE, COLUMN or etc. | VARCHAR(255) |
| IN | **inRECURSE\_**DEPENDENCIES – N or blank = Do not recurse. Y = Apply recursively to depenendencies | VARCHAR(1) |
| IN | **inRECURSE\_DEPENDENTS** – N or blank = Do not recurse. Y = Apply recursively to depenents | VARCHAR(1) |
| IN | **inRECURSE\_CHILD** – N or blank = do not recurse. Set specified privileges for the resource only.  YM = Only apply modification recursively  YC = Make child resources look like this resource recursively  Concept 1: Only select YM to recurse child resources when the resource being set is at the lowest level at which it is safe to recursively set privileges for a given USER or GROUP. If the resource is an intermediate level folder with other branches offshoot from it then only apply the privilege modification to the resource folder using "N" unless all branches are required to have the same privilege settings. "YC" should only be used when you want to reset all privileges for other USERS or GROUPS to "NONE" except for the current USER or GROUP being set.  Concept 2: If the higher level resource folder is set with "YC" then it is not necessary to set this USER or GROUP for branch-level resources. | VARCHAR(2) |
| IN | **inREVOKE\_ALL** – Revoke all privileges for all users and all groups prior to setting the specified privileges.  N or blank=Do not revoke all privileges  Y=Revoke all privileges  This only needs to be done once per Resource Path group. If it user chooses to revoke all, then it must only be done on the first entry for a given Resource Path as it will set the privileges to "NONE" for all users and groups. If this value is set to Y on subsequent calls, it will reset any privileges previously set. | VARCHAR(1) |
| IN | **inNAME** – The user or group name. | VARCHAR(255) |
| IN | **inNAME\_TYPE** – This is either USER or GROUP depending on the NAME field. | VARCHAR(255) |
| IN | **inDOMAIN\_NAME** – The domain the group exists in. | VARCHAR(255) |
| IN | **inENVIRONMENT\_NAME** – The Composite server environment in which to update studio rights for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **inREAD\_ACCESS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inWRITE\_ACCESS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inEXECUTE\_ACCESS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inSELECT\_ACCESS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inUPDATE\_ACCESS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inINSERT\_ACCESS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inDELETE\_ACCESS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inGRANT\_ACCESS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inINITIALIZE** – null or X to indicate access. This field should be ignored. | VARCHAR(1) |
| IN | **inOWNER\_NAME** – The owner of the resource which gets set at the time the privilege is applied. For ENVIRONMENT\_NAME=DEV, the owner should be null so that it does not get set. For other environments, it may be any valid user for that environment such as “admin”. | VARCHAR(255) |
| IN | **inOWNER\_DOMAIN** – The owner domain of the resource which gets set at the time the privilege is applied. For ENVIRONMENT\_NAME=DEV, the owner domain should be null so that it does not get set. For other environments, it may be any valid domain in which OWNER\_NAME is contained for that environment such as “composite”. | VARCHAR(255) |
| IN | **inCOMMENTS** – a general comment. | VARCHAR(1024) |
| OUT | **actionType** – I=insert, U=update, D=delete, S=set inactive | CHAR(1) |
| OUT | **rowsProcessed** – Number of rows process for the action type | INTEGER |
| OUT | **duplicateRows** – Number of duplicate rows found | INTEGER |
| OUT | **duplicateRowMsg** – message for duplicate rows | LONGVARCHAR |
| OUT | **message** – a general message | LONGVARCHAR |

## Database Maintenance: Modify Composite Studio Rights

1. modify\_07\_COMPOSITE\_STUDIO\_RIGHTS **–** This procedure is used to insert/update/delete/inactivate rows in the COMPOSITE\_STUDIO\_RIGHTS table. These are rows that come from the Group\_List sheet in the Resource\_Privileges\_LOAD\_DB.xlsx spreadsheet.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/TableMaintenance/ modify\_07\_COMPOSITE\_STUDIO\_RIGHTS

Example:

STUDIO\_RIGHTS\_ID: 2

ORGANIZATION\_NAME: ORG1

PROJECT\_NAME: Demo

GROUP\_NAME: Demo\_Arch

GROUP\_DESC: Demo Architect

DOMAIN\_NAME: composite

ENVIRONMENT\_NAME: DEMO

ACCESS\_FLAG: X

UNDO\_LOCK: [NULL]

READ\_ALL\_CONFIG: [NULL]

MODIFY\_ALL\_CONFIG: [NULL]

READ\_ALL\_RESOURCES: [NULL]

MODIFY\_ALL\_RESOURCES: [NULL]

READ\_ALL\_STATUS: X

MODIFY\_ALL\_STATUS: [NULL]

READ\_ALL\_USER: [NULL]

MODIFY\_ALL\_USER: [NULL]

CREATEDBY: CMPPRVP

CREATEDDATE: 2015-06-16 12:11:46

UPDATEDBY: CMPPRVP

UPDATEDDATE: 2015-06-16 12:11:46

ISACTIVE: Y

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **performInsertUpdate** – Y or N. Y=yes perform insert or update. N=no do not perform insert or update but display the same messages to see what will be executed. | VARCHAR(1) |
| IN | **setInactive** – Set the ISACTIVE flag to N. Y=if the record exists set it inactive. Overrides performDelete. N/null=do not perform this operation. | VARCHAR(1) |
| IN | **performDelete** – Delete the row if it exists. Y=perform a delete using filters supplied. N/null=do not perform delete but perform insert or update instead. | VARCHAR(1) |
| IN | **inSTUDIO\_RIGHTS\_ID** – This is used only when setting a row inactive or deleting a specific row by its generated STUDIO\_RIGHTS\_ID. | DECIMAL(38,0) |
| IN | **inORGANIZATION\_NAME** – The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **inPROJECT\_NAME** – The project name to insert/update/delete/inactivate. If both setInactive and performDelete are null or N then perform the insert. If the row does not exist then insert. If the row already exists then do nothing. | VARCHAR(255) |
| IN | **inSUBPROJECT\_NAME** – The subproject name to insert/update/delete/inactivate. | VARCHAR(255) |
| IN | **inGROUP\_NAME** – The name of the group to be modified. | VARCHAR(255) |
| IN | **inGROUP\_DESC** – The the description of the group. | VARCHAR(255) |
| IN | **inDOMAIN\_NAME** – The domain the group exists in. | VARCHAR(255) |
| IN | **inENVIRONMENT\_NAME** – The Composite server environment in which to update studio rights for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **inACCESS\_FLAG** – null or X to indicate access. | VARCHAR(1) |
| IN | **inUNDO\_LOCK** – null or X to indicate access. | VARCHAR(1) |
| IN | **inREAD\_ALL\_CONFIG** – null or X to indicate access. | VARCHAR(1) |
| IN | **inMODIFY\_ALL\_CONFIG** – null or X to indicate access. | VARCHAR(1) |
| IN | **inREAD\_ALL\_RESOURCES** – null or X to indicate access. | VARCHAR(1) |
| IN | **inMODIFY\_ALL\_RESOURCES** – null or X to indicate access. | VARCHAR(1) |
| IN | **inREAD\_ALL\_STATUS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inMODIFY\_ALL\_STATUS** – null or X to indicate access. | VARCHAR(1) |
| IN | **inREAD\_ALL\_USER** – null or X to indicate access. | VARCHAR(1) |
| IN | **inMODIFY\_ALL\_USER** – null or X to indicate access. | VARCHAR(1) |
| OUT | **actionType** – I=insert, U=update, D=delete, S=set inactive | CHAR(1) |
| OUT | **rowsProcessed** – Number of rows process for the action type | INTEGER |
| OUT | **duplicateRows** – Number of duplicate rows found | INTEGER |
| OUT | **duplicateRowMsg** – message for duplicate rows | LONGVARCHAR |
| OUT | **message** – a general message | LONGVARCHAR |

## Database Maintenance: Modify Composite User

1. modify\_08\_COMPOSITE\_USER **–** This procedure is used to insert/update/delete/inactivate rows in the COMPOSITE\_USER table. These are rows that come from the User\_List sheet in the Resource\_Privileges\_LOAD\_DB.xlsx spreadsheet.

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/ModifyDBPrivileges/TableMaintenance/ modify\_08\_COMPOSITE\_USER

Example:

USERID: 1

ORGANIZATION\_NAME: ORG1

PROJECT\_NAME: Common

SUBPROJECT\_NAME: SB1

USER\_NAME: user1

USER\_DESC: user desc

DOMAIN\_NAME: composite

ENVIRONMENT\_NAME: DEV

USER\_PWD: password

FORCE\_USER: 1

GROUP\_LIST: group1|composite

USER\_NAME\_LOWERCASE: user1

CREATEDBY: CMPPRV

CREATEDDATE: 2017-05-17 16:57:13

UPDATEDBY: CMPPRV

UPDATEDDATE: 2017-05-17 16:57:13

ISACTIVE: Y

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **datasource** – RECOMMENDED FILTER: the datasource where the privileges are stored. If not parameter is provided the default value is used from defaultValues.datasource. Possible values include:   * EXCEL – Excel spreadsheet which gets uploaded to each DV server. * DB\_LLE – SQL Server database for lower level environments. Basically it is used for testing purposes and developing new functionality. * DB\_PROD – SQL Server production database. This connection should always be used unless developing new functionality. | VARCHAR(255) |
| IN | **performInsertUpdate** – Y or N. Y=yes perform insert or update. N=no do not perform insert or update but display the same messages to see what will be executed. | VARCHAR(1) |
| IN | **setInactive** – Set the ISACTIVE flag to N. Y=if the record exists set it inactive. Overrides performDelete. N/null=do not perform this operation. | VARCHAR(1) |
| IN | **performDelete** – Delete the row if it exists. Y=perform a delete using filters supplied. N/null=do not perform delete but perform insert or update instead. | VARCHAR(1) |
| IN | **inUSERID** – This is used only when setting a row inactive or deleting a specific row by its generated USERID. | DECIMAL(38,0) |
| IN | **inORGANIZATION\_NAME** – The name of the organization such as ORG1 or ORG2. | VARCHAR(255) |
| IN | **inPROJECT\_NAME** – The project name to insert/update/delete/inactivate. If both setInactive and performDelete are null or N then perform the insert. If the row does not exist then insert. If the row already exists then do nothing. | VARCHAR(255) |
| IN | **inSUBPROJECT\_NAME** – The subproject name to insert/update/delete/inactivate. | VARCHAR(255) |
| IN | **inUSER\_NAME** – The name of the user to be modified. | VARCHAR(255) |
| IN | **inUSER\_DESC** – The the description of the user. | VARCHAR(255) |
| IN | **inDOMAIN\_NAME** – The domain the group exists in. | VARCHAR(255) |
| IN | **inENVIRONMENT\_NAME** – The Composite server environment in which to update studio rights for: [DEV, UAT, PROD]. | VARCHAR(255) |
| IN | **inUSER\_PWD** – The user password. | VARCHAR(255) |
| IN | **inFORCE\_USER** – 0 or 1 to indicate access. | VARCHAR(1) |
| IN | **inGROUP\_LIST** – Pipe separated list of group|domain. It may contain a list of pairs separated by spaces. It may not exceed 2048 charaacters. Below the list contains a space separate list of group|domain pairs: E.g. G1|D1 G2|D1 G3|D1  group1|composite  ^---GROUP\_NAME|DOMAIN\_NAME | VARCHAR(2048) |
| OUT | **actionType** – I=insert, U=update, D=delete, S=set inactive | CHAR(1) |
| OUT | **rowsProcessed** – Number of rows process for the action type | INTEGER |
| OUT | **duplicateRows** – Number of duplicate rows found | INTEGER |
| OUT | **duplicateRowMsg** – message for duplicate rows | LONGVARCHAR |
| OUT | **message** – a general message | LONGVARCHAR |

## Group Clean-up: Remove Groups

1. remove\_groups **–** This procedure is used to remove the existence of groups from DV using the free-form list "group\_list\_string". Edit the group\_list\_string here: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/CleanupGroups/group\_list\_string

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/CleanupGroups/remove\_groups

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **indomainName** – Provide the Composite domain in which to validate the group list. This may be a composite domain or ldap domain. e.g. composite, ldap or ldap\_im or whatever the domain name is where the groups reside. | VARCHAR(255) |
| IN | **inExceptionList** – [OPTINOAL] Add an item to the exclusion list to keep from deleting that group. Each item must be appended with a comma including the last one. | LONGVARCHAR |
| IN | **inInclusionList** – [OPTIONAL] Add an item to the inclusion list to test one or more specific groups. Each item must be appended with a comma including the last one. | LONGVARCHAR |
| OUT | **result** – The result of this operation returns a cursor of rows | PIPE ( GroupName VARCHAR(255), DomainName VARCHAR(255), Status VARCHAR(255) ) |

## Group Clean-up: Validate Groups

1. validate\_groups **–** This procedure is used to validate the existence of groups from DV using the free-form list "group\_list\_string". Edit the group\_list\_string here: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/CleanupGroups/group\_list\_string

Location: /shared/ASAssets/BestPractices\_v81/PrivilegeScripts/CleanupGroups/validate\_groups

| **Direction** | **Parameter Name** | **Parameter Type** |
| --- | --- | --- |
| IN | **indomainName** – Provide the Composite domain in which to validate the group list. This may be a composite domain or ldap domain. e.g. composite, ldap or ldap\_im or whatever the domain name is where the groups reside. | VARCHAR(255) |
| IN | **inExceptionList** – [OPTINOAL] Add an item to the exclusion list to keep from deleting that group. Each item must be appended with a comma including the last one. | LONGVARCHAR |
| IN | **inInclusionList** – [OPTIONAL] Add an item to the inclusion list to test one or more specific groups. Each item must be appended with a comma including the last one. | LONGVARCHAR |
| OUT | **result** – The result of this operation returns a cursor of rows | PIPE ( GroupName VARCHAR(255), DomainName VARCHAR(255), Status VARCHAR(255) ) |