**ExxonMobil Chemical Company**

**Release Build Manual**

**Version 1.23**

**May 23, 2022**

`

**Prepared by:**

**ILS Automation Inc.**

Table of Contents

[1 Introduction 4](#_Toc104201668)

[1.1 System cross Reference 4](#_Toc104201669)

[1.2 Installer cross reference 4](#_Toc104201670)

[1.3 Installer Background 5](#_Toc104201671)

[2 Preparation 5](#_Toc104201672)

[2.1 Determine Version for the Release 5](#_Toc104201673)

[2.2 Database 6](#_Toc104201674)

[2.2.1 Determine if the database has changed 6](#_Toc104201675)

[2.2.2 Version Table 6](#_Toc104201676)

[2.2.3 Make a copy of the master database 7](#_Toc104201677)

[2.2.4 Database Create Script 7](#_Toc104201678)

[2.2.5 Database Update Script 9](#_Toc104201679)

[2.2.6 Test the Update Script 10](#_Toc104201680)

[2.2.7 Startup Script 10](#_Toc104201681)

[2.2.8 Manual Database Update Window 12](#_Toc104201682)

[2.3 Ignition 12](#_Toc104201683)

[2.3.1 Project 13](#_Toc104201684)

[2.3.2 Global Project 13](#_Toc104201685)

[2.3.3 DB Manager Project 14](#_Toc104201686)

[2.3.4 UDTs & Scan Classes 14](#_Toc104201687)

[2.4 ILS-Python 14](#_Toc104201688)

[2.5 Release Notes 14](#_Toc104201689)

[2.6 Subversion 15](#_Toc104201690)

[2.7 git Repositories 15](#_Toc104201691)

[2.7.1 ignition-79 15](#_Toc104201692)

[2.7.2 ils-common 15](#_Toc104201693)

[2.7.3 tools 15](#_Toc104201694)

[2.7.4 sfc 15](#_Toc104201695)

[2.7.5 blt 15](#_Toc104201696)

[3 New Site Installer 16](#_Toc104201697)

[3.1 Installer Build 16](#_Toc104201698)

[3.2 Release Verification 16](#_Toc104201699)

[4 Update Installer 16](#_Toc104201700)

[4.1 Installer Build 16](#_Toc104201701)

[4.2 Release Verification 17](#_Toc104201702)

[5 Miscellaneous 17](#_Toc104201703)

[5.1 Cleaning up Stranded SFCs 17](#_Toc104201704)

[5.2 Transferring SFCs 17](#_Toc104201705)

[5.3 Ignition Internal Database 18](#_Toc104201706)

# Introduction

This document provides a step by step guide to releasing software for the ExxonMobil Chemicals project.

For the duration of this manual, SVNRoot refers to Q:\EMChemicals

## System cross Reference

This describes the various Ignition instances and databases on the ILS development system.

|  |  |  |
| --- | --- | --- |
| Service | Database | Purpose |
| EMC Development | XOM-DEVELOPMENT / XOM-DEVELOPMENT-ISOLATION | The install for all active development. This contains lots of test diagrams and SFCs. It also contains the latest version of all windows, UDTs, etc. This also contains the site specific assets but are not considered the master for those assets. |
| EMC | XOM / XOM-ISOLATION | The complete project without any site specific or test assets. |
| EMC Test | XOM-TEST / XOM-ISOLATION | A general test instance that can be used to test installers. |
| APCW |  | Meant for a new hypothetical customer |
| APCW Demo |  | Meant for an IA Demo |

## Installer cross reference

Building an installer requires three files: a bom file, a build file, and a module definition file:

* bom-\*.xml : This contains a specification for each page of the installer. If additional resources are altered, UDTs for example, then a new page may need to be added.
* \*-module.xml : This is a description of the module that will be built and should not have to be changed.
* build-\*.xml : This is a script that gets executed. It builds and signs the module after collecting all of the assets that will be bundled into the module. It needs to be updated for every build to incorporate the correct database update script. If additional resources are changed it may also need to be modified.

The files used for the full installer and the update installer, along with the BRCP legacy installer are described below (all files have extension .xml):

| Product | BOM | Module | Builder | Description |
| --- | --- | --- | --- | --- |
| EMC Installer | bom-emc-install | emc-installer-module | build-emc-installer | Used to install the complete system onto a new site. |
| EMC Updater | bom-emc-update | emc-updater-module | build-emc-updater | Used for an incremental updates. |
| BRCP Installer | bom-brcp | brcp-installer-module | build-brcp-installer | This contains all of site specific assets for every site. This is obsolete as of 1.0 where ownership was transferred to EM for all site assets |

## Installer Background

This section describes the process for building an installer for new ExxonMobil sites and an update installer for existing sites. Neither installer contains any site specific resources but may contain Exxon specific resources.

The update installer assumes that the site is running version 1.0 or newer. The update installer strategy is to deliver all Ignition resources (windows, templates, reports, internal and external scripts, and modules) with each version. We could attempt to determine exactly which resources were changed since the previous release but Ignition’s source code control is not robust enough to do this in a reliable manner. Therefore, rather than risk missing a changed resource, all the resources owned by ILS will be included in each update installer. The partial project will be merged into the site specific project and all duplicate ILS resources will be overwritten. Therefore, if the gateway is at 1.2 and the update installer is for 1.5, because all of the resources are included in the installer, it is not necessary to incrementally update the system.

The database takes a different approach. Because the site database contains a wealth of site specific data, the tables cannot simply be dropped and recreated. ILS maintains rigorous source code control over the database schema and uses SQL Compare to generate focused update scripts. The update scripts for each version are incorporated into a cumulative update script that checks the site version in the version table and then runs the appropriate updates. It updates the version table as updates are applied. In summary, database updates are applied incrementally, but that are all included in each installer.

# Preparation

This section describes steps to follow to prepare resources for both the new site and update installers.

## Determine Version for the Release

Determine the version for the release. The version should be of the form X.YrZ.

X: major version

Y: minor version

Z: revision

Once the version is determined for the release, update the following files:

* Q:/EMChemicals/RELEASES.txt
* Q:/EMChemicals/Release/DATE
* Q:/EMChemicals/Release/VERS
* Q:/EMChemicals/Release/Release Checklist.xlsx

## Database

The SQL\*Server database is at the core of all of the toolkits: SFC, Diagnostic Toolkit, Lab Data, Recipe Data, and custom site specific applications.

A robust scheme for updating the database schema was implemented starting with release 1.2r0. The full installer installs the complete database schema. The update installer does not do any database tasks.

The database update strategy is to check the database version every time the gateway is restarted. Because restarting the gateway is performed after installing an update, the database will be updated immediately after running the update installer and restarting the gateway. The installer copies new database alter scripts to the database folder under the user-lib directory.

### Determine if the database has changed

A release of the toolkits may or may not involve changes to the database. A bug fix release may contain solely changes to Python or a module in which case the database scripts from the previous build are still valid.

Use “SQL Compare” to determine if the current schema used for development has changed. Compare the *XOM-DEVELOPMENT* instance with the most recent *XOM-Development-x.y* instance. If there are not any changes then proceed to section 2.2.6

### Version Table

The Version table stores the version of the database schema. It also records when the database was updated to a specific version. If the database schema has changed then a new record needs to be added to this table. The version of the database schema does not need to match the version of the Symbolic Ai platform.

Graphical user interface, application

Description automatically generated

### Make a copy of the master database

In order to be able to create an accurate update script it is necessary to have an accurate starting point for SQL Compare. Every time a release is make it is important to save a copy of the database. This will become the basis for the next release.



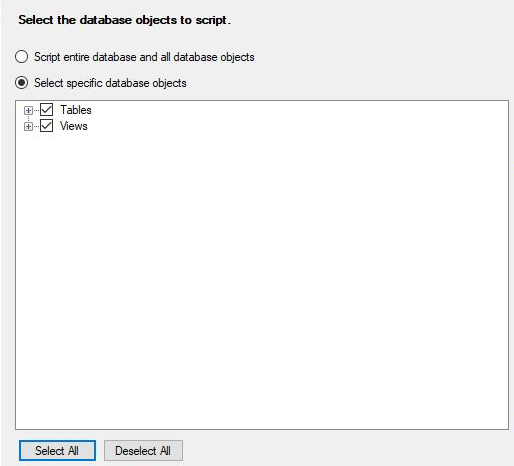
Development will always be done using the XOM-DEVELOPMENT database. Every time a release is made it should be frozen using this procedure:

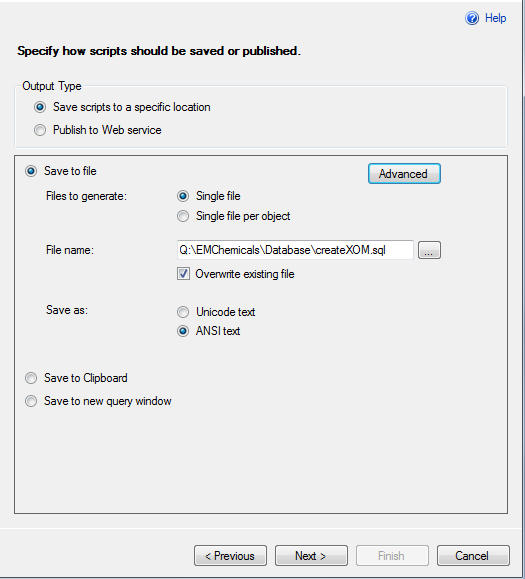
1. Make a backup of XOM-DEVELOPMENT
2. Create a new database instance named XOM-Development-X.Y
3. Restore the backup made in 1 to the database created in 2 making sure to rename the files.
4. Run the database cleanup script, *sweeper.sql*, on the ***new*** database instance. It will delete all of the transient, site specific, and test data and leave a clean instance with only the data necessary for a new site. It will also insert a base record into the *TkSite* table. This record defines the gateway and site startup scripts.

### Database Create Script

Use SQL\*Server to create a database script for the new database instance ***including data***.

The step by step process is:

1. Open SQL\*Server.
2. Right-click on the new database instance created above and select Tasks -> Generate Scripts…
3. On the Choose Objects tab, choose “Select specific database objects”. Unselect the Ignition history tables which may or may not exist in the particular database.
4. Specify the filename as Q:/EMChemicals/Database/createXOM.sql. This is the file that will be included in the full installer



1. Press the “Advanced” button and configure as shown below and press OK

Graphical user interface, text, application

Description automatically generated

1. Press “Next >” twice to generate the database create script.
2. Press “Finish”
3. Copy the Q:/EMChemicals/Database/createXOM.sql to Q:/EMChemicals/Database/create XOM X.YrZ.sql. This file will be stored for archival purposes.

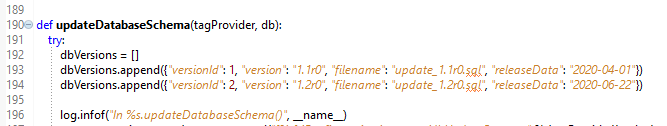
### Database Update Script

Determine if there were any changes to the database structure. If there was a change, then an update script is needed to upgrade the previous database structure to the new database structure. Use SQL Compare to determine if there were changes from *n-1* to *n*. Use SQL Compare to create an update script and save it to Q:/EMChemicals/Database/SQL Compare scripts/XOM-Development-***X***.***Y***r***Z***. Where X, Y, and Z are the major, minor and version numbers. Use this script as the starting point for the update script that will be included in the installer. The final version of the update script should be named update\_***X***.***Y***r***Z***.sql and be placed in Q:/EMChemicals/Database/Update Scripts.

Using Notepad++, clean up the raw script by:

* Consolidating and condensing the header comment to just state the versions and date generated.
* Remove the SET commands at the top
* Remove the GO and PRINT commands
* Terminate each line with a semi-colon.
* If one of the changes is to modify an existing constraint, always delete it first.
* ***Make sure to save the script in ANSI text format (Encoding menu in Notepad++).***

Add a new record to *updateDatabaseSchema()* in *ils.common.startup.py* to specify the information for the new version. The *versionId* should be incremented by 1 for each new release. It should be noted a toolkit release may not necessarily include a database update.



### Test the Update Script

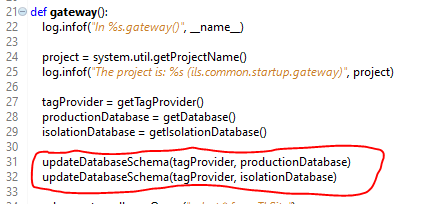
It is important to test the update script because there are several manual steps and various software products.

1. In SQL\*Server, restore the previous database to XOM-TEST
2. Using SQL\*Compare, compare XOM-DEVELOPMENT to XOM-TEST, the results should be the same as obtained above in 2.2.1
3. In Ignition Designer, run the update script against XOM-TEST using the window Test/Database/DB Update Test.
4. In SQL\*Compare, repeat step #2 – the results should be identical. Note that if a view was updated or create then the extended properties will be flagged as a difference because the extended properties for views are not included in the update script – these changes can be ignored.

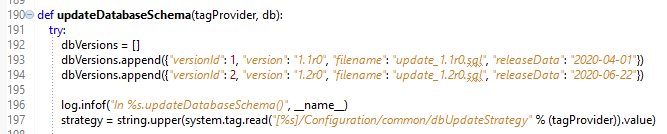
Modify the update script to correct any errors – do not proceed until the update script passes!

### Startup Script

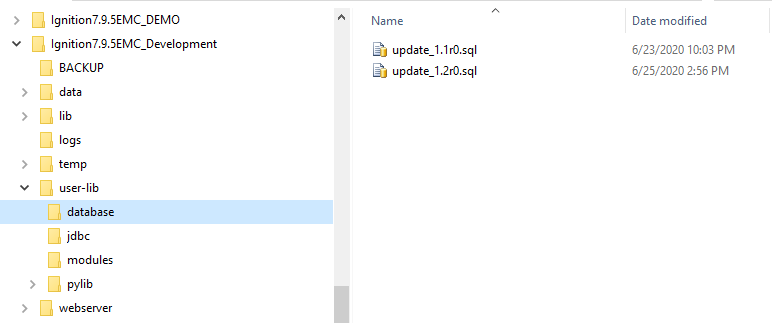
A check was added to the standard gateway startup script that compares the version of the database that is stored in the database to the latest version as configured in Python. The relevant lines of the gateway startup script, *ils.common.startup.py*, are shown below:



The data structure that is used to compare the current database version to the released version is shown below:



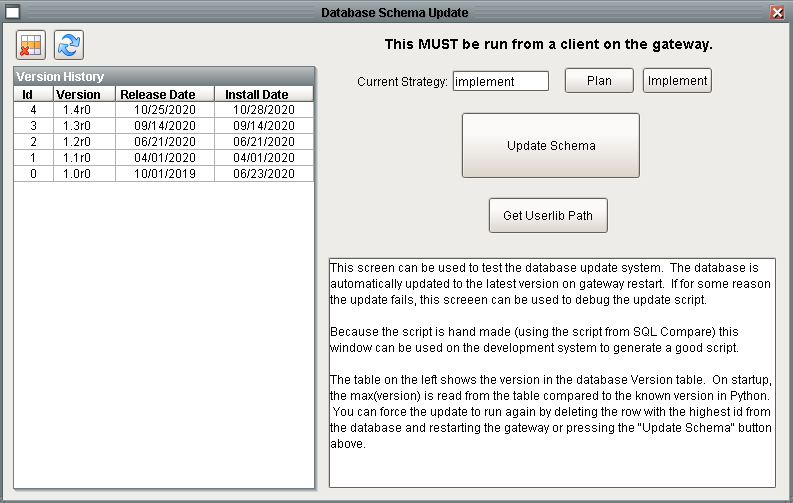
The sql scripts referenced by the *filename* field are copied by the installer into the directory shown below.



The *startup* script reads the current database version, compares it to the versions in the *dbVersions* data structure and runs the SQL commands in the file if the current version is less than the *versionId*. As updates are run, the version table is updated. The commands in the SQL file are executed one at a time from Python by calling *system.db.runUpdateQuery().* One of the challenges that is worth mentioning is that the SQL command file is read by *system.file.readFileAsString()* which takes a fully qualified path to the file. The installer handles installing the SQL command files relative to the current running Ignition. There is a Python method for getting the User Lib path in ils.common.configGateway.py

### Manual Database Update Window

In addition to the automatic update process described above, there is a window for debugging and manually running the database update scripts. The window shown below can be launched from a client using the Admin -> Configuration -> DB Schema Update.



The table on the left shows the version history of the database. There isn’t any visibility to what version of the database should be. Running the update in “Plan” mode allows you to make a dry run of the update. “Implement” mode will actually execute the SQL commands.

The update script is just a text file of SQL commands. The file can be edited and updated if it fails for any reason. Use this window to rerun the update scripts without having to restart the gateway. The engine that runs the SQL commands uses error handling to catch errors and continue processing. If some updates failed resulting in the version table falsely indicating the database version, you can force the script to rerun by deleting the version record from the table. The update logic queries the highest version in the table. For example, deleting version 2 (1.2r0) from the table above will not cause the update script to run because the highest version is still 4 (1.4r0).

## Ignition

This captures changes in the main project, the global project, and in the DB Manager project. Start the “EMC Development” Ignition image.

### Project

* Start the “EMC Development” Ignition image.
* In Designer:
  + Open the XOM project
  + Update the version and date on the Admin/Version window.
  + Save the project
* In the Gateway web page:
  + Make a gateway backup
* In Designer:
  + Open the “XOM” project
  + Export project resources for the new site / full installer:
    - Export the project using File -> Export to *SVNRoot*/Ignition/XOM\_Core\_***X***.***Y***r***Z***.proj. Where X, Y, and Z are the major, minor and version numbers.
    - Unselect the following:
      * All diagnostic applications / Families / Diagrams.
      * All Test & Site specific windows, reports, queries.
      * Scripts/Test (not sure if this still exists)
  + Export project resources for the existing site / incremental installer:
    - Export the project using File -> Export to *SVNRoot*/Ignition/XOM\_Update\_***X***.***Y***r***Z***.proj, where X, Y, and Z are the major, minor and version numbers.
    - Unselect all of the same things mentioned above.
    - In addition, unselect:
      * *Project Properties* folder.
      * *Scripts / Designer Props*.
  + Using the Windows File Manager:
    - Copy XOM\_Update\_***X***.***Y***r***Z***.proj to XOM\_Update.proj
    - Copy XOM\_Core\_***X***.***Y***r***Z***.proj to XOM.proj

### Global Project

* Switch back to the “EMC Development” Ignition image…
* In Designer:
  + Open the XOM project
  + Select File -> Export Global and select:
    - Alarm Pipeline
    - Global scripts
  + Save the global project to: *SVNRoot*/Ignition/XOM\_Global.proj and to XOM\_Global\_***X***.***Y***r***Z***.proj. (There may be holes in the numbering if some releases do not have changes to the global project.)

### DB Manager Project

There are no site-specific changes to the DB Manager project, therefore the entire project can be supplied in the update installer.

* Update Version – In Designer, update the version on the Admin/Version window.
* Project – In the gateway, export DB Manager project to ***SVNRoot\***Ignition

### UDTs & Scan Classes

Export UDTs and scan classes. I’m not sure how updated / new UDTs will be distributed in the update installer. I think that if there are changes then it can be deployed just like it was in the regular installer. The page will need to be added back into the *bom-update.xml* file.

* In Designer, export all UDTs to the common resource folder (***SVNRoot***\Ignition\XOM UDTs.xml)
* Export all scan classes to ***SVNRoot\***Ignition\scanclasses.xml

## ILS-Python

Update the version files; these are displayed in the Vision about screen and in the wrapper log on startup. Commit and push changes made in the ILS\_Python repositories to GIT. The XOM Python repository is site specific and is NOT distributed.

* Update ils.common.version.py
* Update ils.demo.version.py.
* GIT Commit & Push

## Release Notes

The release notes are manually generated using information from the TRAC report title *“Release Report for EMC Toolkit X.YrZ”*.

* In Windows File Manager, Open ***SVNRoot***\Release\ReleaseNotes.docx
* In a browser, open the EMC TRAC, select View Reports, copy the most recent release report, update the query and header to contain the milestone number.
* Run the report
* Make the Chrome window about half of the display width.
* Select the table, copy and paste into the word document. Resize and reformat the table as needed. Use *Merge Cells* to get rid of the crap at the end of the row.
* Save the document as a PDF to ***SVNRoot***\Release\ReleaseNotes.pdf. This will be picked up and included in the installer.

## Subversion

Check in all documentation that are stored in the EM Chemicals Subversion repository.

## git Repositories

Visit each *git* repository in turn. Update and build.

### ignition-79

Make sure that the libraries correspond to the latest version of Ignition. If not, update the repository from maven.

* cd ${GIT\_REPO}/ignition-79
* git pull

### ils-common

This doesn’t change very often.

* cd ${GIT\_REPO}/ils-common
* git pull
* build the ils-common jar

### tools

This contains the testing framework which is used internally or as part of a distribution-for-test, but are not included in the production release.

* cd ${GIT\_REPO}/tools
* git pull

In eclipse:

* Switch to the Tools workspace
* Project TF\_Build - run build-tf.xml.

### sfc

This contains the ils-sfc module, chart migration.

* cd ${GIT\_REPO}/sfc
* git pull

In eclipse:

* Switch to the “Eclipse” workspace.
* Project ILS\_Common - run build-ils-common-jar.xml
* Project sfc-build - run build-ils-sfc.xml and build-migrator.xml

### blt

This builds the blt module.

* cd ${GIT\_REPO}/blt
* git pull

In eclipse:

* Switch to the eclipse workspace
* Make sure ils-common and ils-python are loaded.
* Project BLT\_Build:
* Run build-blt-javadoc (This Builds the java docs for BLT which are embedded in the BLT module and are accessible from the gateway web page, config, modules. It will have lots of errors which can be ignored)
* Run build-blt.xml

# New Site Installer

This installer does not contain any site specific resources. This should be used to configure Ignition on a new gateway. It will instal a new project and a new database. It will completely overwrite any existing data in either.

## Installer Build

This section describes how to build the installer module for a new site.

To build the installer:

In a bash window:

* cd ${GIT\_REPO}/installer
* git pull

In eclipse:

* Switch to the installer workspace.
* Load all projects in the repository into Eclipse. In AI-build, execute build-ai-jars.xml. This updates the installer code. (Only needed if the installer code has changed.)
* In the *emc* project, execute *build-emc-installer.xml*. The script:
  + Builds the installer module
  + Signs it
  + Copies it to u:/work/EMC/installers

## Release Verification

Execute the tests in the ***Release Test Specification*** for the full installer.

# Update Installer

This installer will perform an incremental update from the immediate previous version.

## Installer Build

To build the installer:

In a bash window:

* cd ${GIT\_REPO}/installer
* git pull

In eclipse:

* Switch to the installer workspace.
* Load all projects in the repository into Eclipse. In AI-build, execute build-ai-jars.xml. This updates the installer code. (Only needed if the installer code has changed.)
* In the emc project, execute build-emc-updater.xml. The script:
  + Builds the update installer module
  + Signs it
  + Copies it to u:/work/EMC/installers

## Release Verification

Execute the tests in the ***Release Test Specification*** for the update installer.

# Miscellaneous

This section addresses various miscellaneous issues.

## Cleaning up Stranded SFCs

Occasionally we have found stranded SFCs in the gateway. This should not be confused with issues with the SFC hierarchy or catalog of charts in the SQK\*Server database. This issue is with the storage of SFCs in Ignition’s internal database. This process will preserve all of the charts that are in Designer and delete the hidden charts that are in the internal database but cannot be seen in the designer. This process generally was caused by an error in the ILS installer. This step requires editing the internal database and should be done with utmost care.

1. Download “DB Browser for Sqlite”
2. Open the internal database in “e:program files/Induct…/data/db/config.idb”
3. Export Charts using Designer
4. Delete charts from the designer
5. Shut down the gateway
6. Use the Sqlite browser to delete all remaining sfc resources in table PROJECT\_RESOURCES.
7. Restart the gateway
8. Import the charts exported above.

## Transferring SFCs

This procedure is used when gathering the latest SFCs from a site to include into the installer or when transferring SFCs without creating an installer. This is for the version of SFCs that does not support the export and import of recipe data. It assumes that all of the SFCs from the target system are to be copied to the source system.

**From the Source system (order is not important):**

1. In a client, open the Recipe Data Browser, select the top of the tree and press export.
2. In the designer, from File, select Export Global

**At the target system (order IS important):**

1. In the designer delete all traces of the SFCs.
2. In SQL\*Server, examine the SfcChart table for any stranded charts. Manually delete any remaining SFCs. Browse the recipe data tables and make sure there are no stranded data. These should have been deleted by cascade delete constraints.
3. In a client, import the recipe data
4. In the designer, import the global project

## Ignition Internal Database

The internal database can be viewed using the SQLite DB Browser.

Some useful commands are:

select distinct RESOURCETYPE from PROJECT\_RESOURCES

select \* from PROJECT\_RESOURCES

where RESOURCETYPE = "blt.family"

select \* from PROJECT\_RESOURCES

where RESOURCETYPE = "blt.family"