Revision Date: August 29, 2013

**Purpose:** The Database Refresh utility enables BU's to initiate refresh of a selected database using the most recent database backup from another database on another instance. The utility is used primarily to refresh a test database from a production database.

Also see: **Technical Overview** for an explanation of how the utility works.

**Troubleshooting** for steps to fix a refresh when there was insufficient disk available

# Setup Instructions

The GDTS DBA installs the Self-serve Database Refresh Utility on each instance requested as follows:

1. Create login GDTSRefreshDB on the Source instance and create the necessary linked server on the destination instance by:

* Connect to **psql01\_pgmaint\pgmaint as yourself** and execute the following procedure from database cargilldbamaint providing parameter @source and parameter @destination where @source is the production server and @destination is the instance where the database will be refreshed :  
    
   exec cargilldbamaint.dbo.csp\_setup\_GDTSRefreshDB @source='<source instance'>, @destination='destination instance'>
* The procedure should complete with no errors and should print out information similar to:

|  |
| --- |
| Source is DEHAMD005M  Destination is DEHAMD008M  Agent account on [DEHAMD008M] is [EU\dehamd008m\_sql]  Added Linked server [DEHAMD005M] on [DEHAMD008M]  Had to add MAINTUSER to ip\_auth\_chk table  created GDTSRefreshDB as login on DEHAMD005M  Added Login GDTSRefreshDB as a user in msdb on DEHAMD005M  Granted db\_datareader to GDTSRefreshDB in msdb on DEHAMD005M  Granted VIEW ANY DEFINITION on DEHAMD005M  Added Login GDTSRefreshDB as a user in cargilldbaip on DEHAMD005M  Granted IP\_PROFILE\_GDTS to GDTSRefreshDB in cargilldbaip on DEHAMD005M  Granted IP\_ROLE\_GDTS\_IP\_DBA to GDTSRefreshDB in cargilldbaip on DEHAMD005M  Added GDTSRefreshDB to ip\_logins in cargilldbaip on DEHAMD005M  Added GDTSRefreshDB to ip\_ds\_override in cargilldbaip on DEHAMD005M |

* Verify that a linked server was created on the destination instance and that it contains a row in the security tab for 'sa' and a row for the owner of the agent services. Both should be using GDTSRefreshDB as the remote user.

1. Determine the location on the **destination server** to hold the copied backups from the source server

* Create this directory on the **destination server**.  
   Usually the location should be at the same level as the normal database backup. For example  
   if the destination backups are stored at "G:\MSSQL.1\MSSQL\Backup\DB"  
   Create directory at "G:\MSSQL.1\MSSQL\Backup\RefreshDB"

1. Setup privileges on the **source instance** to allow the destination instance to copy the backups

* Creating a share on the **source instance** and grant privileges:  
    
   Use Computer management, Shared Folders, Shares, right click and select ‘create a share’

|  |  |
| --- | --- |
| Enter the folder path the holds the backups for the instance  Click Next |  |
| Enter the share name and a short description  Use SelfServeRefresh and if that is already created SelfServeRefresh1, etc  Click Next |  |
| Set the permissions  Select ‘Use custom permissions’  click Custom |  |
|  |  |
| Place check marks in Full Control, Change and Read  Click on OK |  |
| Click on Finish |  |
| Click on Finish  And then click on Close |  |
| Drill down to the directory.  Right click on the directory and choose 'Properties' and then choose ‘Security'  Click on 'Edit'  If there is no 'Edit' button click on 'Add' |  |
| Click on the Add button |  |
| Add the service owner for the destination instance and click ok |  |
| Grant access to the destination service owner which was added in the previous step.  Grant the destination service owner Read & Execute, List Folder contents and Read  Click ok |  |

1. Create and populate Cargill\_DBA and tempDB objects on the **destination instance**.

* Create the Cargill\_DBA stored procedure:

Load \\ADMPLS233M\GDTSDATA\\_private\SQL Server\GDTS\_setup\SelfServe\_BU\_DB\_refresh\**Refresh\_procedure.sql** into a query window connected to the **destination instance**. The script properly sets the database context before creating the stored procedure. The script creates procedure csp\_selfServe\_Refresh

* Create utility tables and insert row in tablespurge\_meta on the **destination instance**:  
    
  Create meta table **dbo.selfServe\_Refresh\_meta** in Cargill\_DBA, audit table **dbo.selfServe\_Refresh\_audit** in both Cargill\_DBA and in tempdb, and insert a row in Cargill\_DBA.dbo.tablespurge\_meta for the audit table by running this stored procedure without any parameters on the destination instance:  
   EXEC Cargill\_DBA.dbo.csp\_selfServe\_Refresh
* Verify that these tables were created by executing the following on the **destination instance**:  
   select \* from Cargill\_DBA.dbo.selfServe\_Refresh\_meta  
   select \* from Cargill\_DBA.dbo.selfServe\_Refresh\_audit  
   select \* from tempdb.dbo.selfServe\_Refresh\_audit
* Verify that a row was inserted into Cargill\_DBA.dbo.tablespurge\_meta by retrieving the information from the tablespurge\_meta and from the audit table on the **destination instance**:  
   SELECT \* FROM Cargill\_DBA.dbo.tablespurge\_meta  
   SELECT \* FROM Cargill\_DBA.dbo.selfServe\_Refresh\_audit

1. If the database to be refreshed does not already exist on the **destination instance**, create it by restoring a copy using a backup from the source instance.
2. Insert the information in the meta table for the database to be refreshed on the **destination instance**.

* Determine the values for the following parameters ( see the **Technical Overview** section for explanation of the columns)

src\_dbNm --source database name

src\_instanceNm --source instance name

dstn\_dbNm -- destination database name

dbowner -- (database owner if not provided the table defaults 'sa' )

dbRecoveryModel --(databse recovery mode -- if not provided the table defaults to 'FULL')

destBackupLoc --destination backup location

src\_backupShare --source share name

deleteCopiedFullBackups --(if not provided the table defaults 'N' )

deleteCopiedDiffBackups --(if not provided the table defaults 'N' )

FreeDiskRequired\_0\_200gb\_pct --(if not provided table defaults to 20)

FreeDiskRequired\_200\_500gb\_pct --(if not provided table defaults to 10)

FreeDiskRequired\_over\_500gb\_gb -- (if not provided table defaults to 50)

FreeDiskBackupRequiredMB --(if not provided table defaults to 1024)

Example inserting just the minimal information needed:

INSERT INTO Cargill\_DBA.dbo.selfServe\_Refresh\_meta

(src\_dbNm, src\_instanceNm, dstn\_dbNm , destBackupLoc, src\_backupShare )

VALUES

('SSISTutorial',

'ADMPGS040M\GDTSDRTEST',

'SSISTutorial',

'G:\SQL2008R2CS\MSSQL\BACKUP\RefreshDB',

'\\ADMPGS040M\SelfServeRefresh\')

1. Set up Job 'SelfServe - Refresh' on the **destination instance**.

* If job 'SelfServe - Refresh' does not exist on **destination instance**:  
    
  Load \\ADMPLS233M\GDTSDATA\\_private\SQL Server\GDTS\_setup\SelfServe\_BU\_DB\_refresh\**selfServe\_Refresh\_job.sql** into a query window connected to the destination instance.   
  Replace xxdatabasexx with the name of the database to be refreshed and then execute the script. After the job is created, update the job to add an output file in the appropriate location to the job steps. The output file should be directed to the following replacing <instance> with the instance portion of the name and <database> with the name of the database to be refreshed.  
   d:\cargill\logs\<instance>\Refresh\_<database>.txt
* If job 'SelfServe - Refresh' does exist on **destination instance**:

Add job step immediately before 'Final Step' and name the step 'Refresh <database name>' replacing <database name> with the name of the database to be refreshed.

Job step should contain the following (replace <database> with the database name)  
 exec Cargill\_DBA.dbo.csp\_selfServe\_Refresh @dstn\_dbNm= '<database>'

On failure, the job step should go on to the next step.  
Update the job step to add an output file in the appropriate location to the job step. The output file should be directed to the following replacing <instance> with the instance portion of the name and <database> with the name of the database to be refreshed   
 d:\cargill\logs\<instance>\Refresh\_<database>.txt

1. Execute the job on the **destination instance**. (It will NOT refresh the database at this time since the trigger table is empty)

* When the job runs it will create the trigger table in the database to be refreshed if it does not already exist (GDTS\_rw#u7#WQ\_refresh). Verify the trigger table was created by executing the following on the destination instance connected to the database to be refreshed:  
   select \* from dbo.GDTS\_rw#u7#WQ\_refresh

1. If the BU requests additional items to be done after the refresh (for example adding users) add a job step after the refresh step for the particular database (call the job step Post Refresh <database> ) .
2. If the refresh process should execute on a daily or weekly schedule, set up a job to insert a row in the trigger table on the destination instance.   
    Name the job 'SelfServe <database> Initiate Refresh'   
    Job should have one step which contains:   
    insert into [<dbname>].dbo.GDTS\_rw#u7#WQ\_refresh values('go', null)  
    Schedule the job based on the BU requirements for the refresh
3. When the first refresh has been executed:  
   * The procedure will verify the destination directory exists and it will then add rows to the Cargill\_DBA.dbo.filesdel\_meta for the directory listed in the destBackupLoc parameter (location for the copied backups on the destination server). This will be default to 10 days. To see the rows were added to the table, execute the following on the destination instance:  
      Select \* from Cargill\_DBA.dbo.filesdel\_meta  
     If the number of days to keep the backups needs to be changed, manually update the filesdel\_meta table.
4. If LiteSpeed is installed on the source instance (there is a database named LiteSpeedLocal) but not on the destination instance, the DBA will need to copy the LiteSpeed executable to c:\cargill\bin on the destination server. Should find the executable in the **C:\Program Files\Imceda\LiteSpeed\SQL Server\Engine** directory on the source server The executable will be named:  
     
    SQLLiteSpeed.exe, SQLLiteSpeedx32.exe or SQLLiteSpeedx64.exe  
     
    If there are multiple executables, **ONLY** copy one of the executables.
5. Send instructions to the BU on how to operate the self-serve Refresh Utility. Text for the communication appears on the following page.

Instructions for the BU (paste into an email using the Paste Special > Formatted Text (RTF) option):

|  |
| --- |
| Instructions to use the GDTS self-serve database refresh utility:  Step 1: When needed, trigger a refresh:  To refresh a particular database simply place any value in the refreshstep column of table dbo.GDTS\_rw#u7WQ\_refreshin the particular database to be refreshed. Example:  INSERT INTO dbo.GDTS\_rw#u7#WQ\_refresh VALUES ('start refresh',null)  Within 5 minutes a database refresh will be initiated. After the refresh completes the trigger table will again be empty. To initiate another refresh, rerun the INSERT statement.  **NOTES**  The refresh utility will always first verify there is enough space on disk to complete the refresh without over-filling drives. The refresh will not run if there is insufficient space. If there are problems with the refresh the tempdb..selfServe\_Refresh\_audit will contain information about the error.  **HANDLING ERRORS**  When a refresh fails to happen, check the contents of column **error\_msg** in table **dbo.GDTS\_rw#u7WQ\_refresh**. If a value is present, correct the problem as explained in the column. Then NULL out the **error\_msg** column value using:  UPDATE dbo.GDTS\_rw#u7#WQ\_refresh SET error\_msg = NULL  The NULL value in column error\_msg re-enables the refresh utility which will then attempt another refresh within 5 minutes.  **RESPONSIBILITIES**  For dedicated servers, it is the responsibility of the BU application DBA that uses the self-serve database refresh utility to manage space for database refreshes.  The refresh utility first checks available disk space before performing a refresh. It will not perform the refresh if it detects that low free disk space on the drive where the backup file will be copied or on the drives where the data files are located.  The backup files will be aged off on the destination server. If the BU wants to keep the files on the destination server, they will need to move the backup files to another location. |

# Technical Overview

This is a self serve process to allow a BU to refresh one of their own database(s) on a dedicated server or on a GDTS shared server.

Initial setup is done by running procedure csp\_setup\_GDTSRefreshDB from psql01\_pgmaint\pgmaint:

csp\_setup\_GDTSRefreshDB @source='<source instance'>, @destination='destination instance'>

* This procedure creates the GDTSRefreshDB login on the source instance with the appropriate IP information
* This procedure inserts a row into table addbrass.dbo.GDTSRefreshDB\_location for each instance which is using the refresh procedure. The table signifies which instance has the GDTSRefreshDB login and which instance has the linked server.
* Adds user GDTSRefreshDB to MSDB as datareader on the source instance:

|  |
| --- |
| USE [msdb]  GO  CREATE USER [GDTSRefreshDB] FOR LOGIN [GDTSRefreshDB]  GO  EXEC sp\_addrolemember N'db\_datareader', N'GDTSRefreshDB'  GO |

* Grants privileges to GDTSRefreshDB on the source instance:

|  |
| --- |
| USE [master]  go  grant VIEW ANY DEFINITION to GDTSRefreshDB  go  If (select count(\*) from master.dbo.sysobjects where name ='xp\_restore\_filelistonly') > 0  BEGIN  CREATE USER [GDTSRefreshDB] FOR LOGIN [GDTSRefreshDB]  grant execute on xp\_restore\_filelistonly to GDTSRefreshDB  END |

* Creates a linked server on the destination instance

Execute one sql script to create one stored procedure in Cargill\_DBA and one script to create a job which will execute the refresh(s). As part of setup the stored procedure creates a meta table in Cargill\_DBA, an audit table in Cargill\_DBA and in tempdb, inserts a row in the Cargill\_DBA.dbo.tablespurge\_meta table for cleanup of the audit table, and inserts rows in Cargill\_DBA.dbo.filesdel\_meta to handle cleanup of the destination directory for the copied backups defaulting to keep backups for 10 days. If this is correct the DBA should change the purge days. The procedure also creates a trigger table (GDTS\_rw#u7#WQ\_refresh) in the destination database if needed.

If LiteSpeed is installed on the source instance but not on the destination instance, the DBA will need to copy the LiteSpeed executable to c:\cargill\bin on the destination instance. Should find the executable in the **C:\Program Files\Imceda\LiteSpeed\SQL Server\Engine** directory on the source server. The executable will be named:

SQLLiteSpeed.exe, SQLLiteSpeedx32.exe or SQLLiteSpeedx64.exe

Processing refreshes the destination database using the latest backup from the source instance. The destination database must already exist on the destination instance and must contain the same number of data and log files. (This is because it uses the destination database to determine where to locate the data and log files.)

The destination database must be in 'READ\_WRITE' mode.

When the BU wants to refresh the destination database, they insert a row in the trigger table. The next time the **'SelfServe – Refresh**' job executes, the process will try to do the refresh. If a database refresh fails the job goes on to the next step. The last step will fail the job if any of the individual database refreshes has failed.

The refresh will not be done if there is not sufficient disk available for refreshing the database.

The procedure uses the previous 24 hours of information from the Cargill\_DBA.dbo.DiskDriveUsage table to determine the disk size, free disk and used disk. The process applies the thresholds to the information retrieved from the Cargill\_DBA.dbo.DiskDriveUsage table to determine the minumum amount of disk that must be in available after the refresh. The process uses the data and log file sizes from the backup files to compute how much disk will be needed for the particular restore. If after these computations there is not sufficient disk available, the refresh will not be attempted. If the refresh cannot be done because of insufficient disk, the trigger table in the destination database is updated with this information.

If for some reason the refresh could not be completed (for example the backup could not be found or the dbowner parameter contains a login that is not on the destination server) the refresh for that database is skipped and the trigger table in the destination database is updated. Column error\_msg will contain the reason why the refresh could not be completed. The BU must correct the problem, clear out the error\_msg column and then the next time the job runs, the refresh will be completed. The error\_msg column will contain the reason as well as code to clear the column.

In this situation the Cargill\_DBA.dbo.selfServe\_Refresh\_auditand the tempdb.dbo.selfServe\_Refresh\_audit tables will also be updated with the refresh issue.

The last step reviews the Cargill\_DBA.dbo.selfServe\_Refresh\_meta table and if it finds and rows with the refresh\_status ='Fail', the job will be failed.

After the refresh completes, the process will fix any broken users, set the database owner and set the recovery model.

If a BU would like additional work done after the refresh (for example adding users to the database), the DBA will need to add a step after the particular database refresh step with this information.

**Objects required (setup creates the following objects)**:

* Procedure **csp\_selfServe\_Refresh** This procedure is located in Cargill\_DBA and does the refresh work. It has one required parameters which is the destination database. The other parameters which the procedure are stored in the Cargill\_DBA.dbo.selfServe\_Refresh\_meta table. The procedure can be executed without parameters and it will print out instructions.
* Table **Cargill\_DBA.dbo.selfServe\_Refresh\_meta** This table is used by the refresh procedure and contains the refresh parameters for each database being refreshed. There is an additional column (refresh\_status) which is updated during the process. If this column contains a 'Fail' the refresh job will break itself as the last step. (If any of the individual database refreshes fail, the job continues so that one fail does not stop all the refreshes.)
* Table **Cargill\_DBA.dbo.selfServe\_Refresh\_audit** This table is used by the refresh procedure to log the refresh events. A copy of this table is kept in tempdb so that any user can view the audit log. The table in Cargill\_DBA contains all events while the table in tempdb only contains a summary of the events that a user by be interested in. The following sql can be used to query the tables on the destination instance:  
   select \* from Cargill\_DBA.dbo.[selfServe\_Refresh\_audit] order by audit\_id

select \* from tempdb.dbo.[selfServe\_Refresh\_audit] order by commit\_ts

* Job **SelfServe – Refresh** This job contains one step for each database refresh. In addition there is a 'Final Step' which checks the refresh\_status column in the Cargill\_DBA.dbo.selfServe\_Refresh\_metaand if it finds any 'Fail' it will cause the job to fail. If the BU was additional work done after the refresh, the DBA will add an addition job step after the particular database refresh step with this additional work.

The **Cargill\_DBA.dbo.selfServe\_Refresh\_meta** table contains the parameters used by the process:

|  |  |
| --- | --- |
| src\_dbNm | This is the name of the source database on the source instance. A check constraint is used to disallow the following databases from being inserted: master, model, msdb, Cargill\_DBA, cargilldbaip, tempdb, Litespeedlocal |
| src\_instanceNm | This is the source instance |
| dstn\_dbNm | This the database which will be refreshed on the destination instance. A check constraint is used to disallow the following databases from being inserted: master, model, msdb, Cargill\_DBA, cargilldbaip, tempdb, Litespeedlocal |
| dbowner | The process will set the database owner to this value. The table defaults to 'sa if not provided |
| dbRecoveryModel | This process will set the recovery model to this value after the refresh. The table defaults to 'FULL' if not provided |
| destBackupLoc | This is the directory where the backups will be copied to on destination. The process will insert rows in Cargill\_DBA.dbo.filesdel\_meta for this location |
| src\_backupShare | This is the share name which was created on the source server |
| deleteCopiedFullBackups | Flag to signify if Full backup copied to destination should be deleted immediately after refresh. The table defaults this to 'N' if not provided and lets the files delete process clean up the backups copied to the destination server. This column should contain a 'Y' if space is very tight on the destination. When column contains 'N' it may save copying over the full backup if subsequent refreshes will use the same backup. |
| deleteCopiedDiffBackup | Flag to signify if Differential backup copied to destination should be deleted immediately after refresh. The table defaults this to 'N' if not provided and lets the files delete process clean up the backups copied to the destination server. This column should contain a 'Y' if space is very tight on the destination. |
| FreeDiskRequired\_0\_200gb\_pct | Determines the amount of free disk that must be available after the refresh if disk/mount point size for data or log files is between 0 and 200 gb. This value is applied as a percent. If value is not provided it defaults to 10. |
| FreeDiskRequired\_200\_500gb\_pct | Determines the amount of free disk that must be available after the refresh if disk/mount point size for data or log files is between 200 and 500 gb. This value is applied as a percent. If value is not provided it defaults to 20. |
| FreeDiskRequired\_over\_500gb\_gb | Determines the amount of free disk that must be available after the refresh if disk/mount point size for dataor log files is over 500 gb. This value is in GB's. If value is not provided it defaults to 50. |
| FreeDiskBackupRequiredMB | Determines the amount of free disk that must be available after the the backup file is copied to the destination and unzipped (if file is gzipped). This value is in MB's. If value is not provided it defaults to 1024. |
| refresh\_status | Column is populated during the refresh process. Used by the refresh job to determine if the job should fail itself. |

# Additional Notes:

The refresh process cannot be used with for the system databases, tempdb, the GDTS databases (Cargill\_DBA or cargilldbaip) or the LiteSpeed database**.**

If the BU wants to use a more recent backup, they will need to contact GDTS and request a backup be executed and then the process will use this new backup.

# Troubleshooting

The process looks at the last 24 hours when checking the amount of available disk (uses the information in Cargill\_DBA.dbo.diskdriveusage table). What may happen is the refresh fails because there is not sufficient disk and a DBA gets involved and needs to free up disk. Even after freeing up disk the refresh will still fail because the information in the Cargill\_DBA.dbo.diskdriveusage table which the process review is the previous 24 hours (which may include the time frame when there was not adequate free disk).

Work around:

Back up the rows in the diskdriveuage table which contain information about insufficient disk to a temporary table, delete the rows from the diskdriveusage table, clear the error message from the trigger table, do the refresh, put the rows back into the diskdriveusage table, delete the temporary table.

Example: refresh is failing because the D drive has less then 60 Gb free during the previous 24 hour period

Backup the rows:

Use Cargill\_DBA

select \* into diskdriveusage\_bak from diskdriveusage where capturedttm > getdate()-1

and driveorpath='D:' and freemeg < 60000

Delete the rows

delete diskdriveusage where capturedttm > getdate()-1 and driveorpath='D:'

and freemeg < 60000

Clear the message from trigger table

update [<databasename>].dbo.GDTS\_rw#u7#WQ\_refresh set error\_msg=null

Execute the refresh job

Put the rows back into the diskdriveusage table:

Use Cargill\_DBA

insert into diskdriveusage select \* from diskdriveusage\_bak

Drop the temporary table

drop table diskdriveusage\_bak