**DB Refresh by using DD Boost**

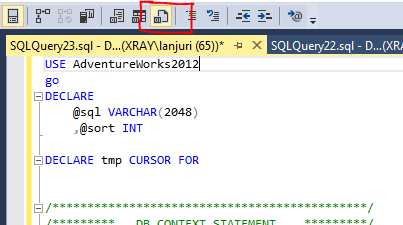
* Before start the DB refresh, please make sure that we have enough free space on drive to do DB refresh
* Take the destination database backup, if latest backup is not available. Please execute the below script to find the latest backup date and backup path.
* Replace the DB\_name with required database name in the below script.

|  |
| --- |
| SELECT  CONVERT(CHAR(100), SERVERPROPERTY('Servername')) AS Server,  msdb.dbo.backupset.database\_name,  msdb.dbo.backupset.backup\_start\_date,  msdb.dbo.backupset.backup\_finish\_date,  msdb.dbo.backupset.expiration\_date,  CASE msdb..backupset.type  WHEN 'D' THEN 'Full'  WHEN 'I' THEN 'Differential'  WHEN 'L' THEN 'Log'  END AS backup\_type,  msdb.dbo.backupset.backup\_size,  msdb.dbo.backupmediafamily.logical\_device\_name,  msdb.dbo.backupmediafamily.physical\_device\_name,  msdb.dbo.backupset.name AS backupset\_name,  msdb.dbo.backupset.description  FROM msdb.dbo.backupmediafamily  INNER JOIN msdb.dbo.backupset ON msdb.dbo.backupmediafamily.media\_set\_id = msdb.dbo.backupset.media\_set\_id  WHERE (CONVERT(datetime, msdb.dbo.backupset.backup\_start\_date, 102) >= GETDATE() - 7) and database\_name='DB\_Name'  ORDER BY  msdb.dbo.backupset.backup\_finish\_date desc |

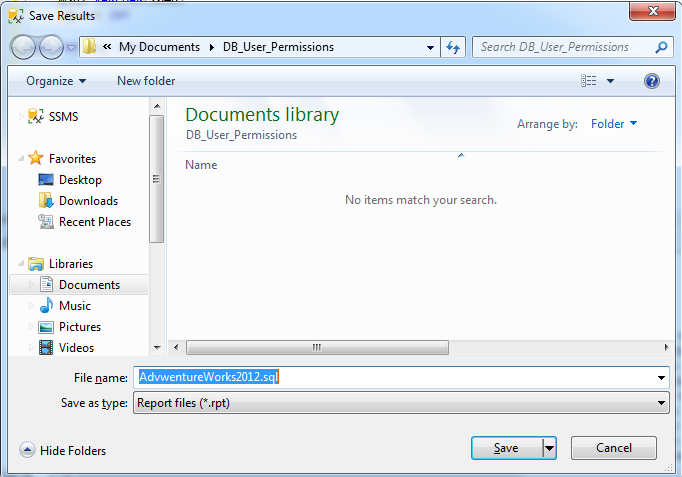
* If destination database backup is not available then take database full backup.
* Take the backup of users permissions by using the below script on destination database.

|  |
| --- |
| USE DBNmae  go  DECLARE  @sql VARCHAR(2048)  ,@sort INT  DECLARE tmp CURSOR FOR  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\* DB CONTEXT STATEMENT \*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  SELECT '-- [-- DB CONTEXT --] --' AS [-- SQL STATEMENTS --],  1 AS [-- RESULT ORDER HOLDER --]  UNION  SELECT 'USE' + SPACE(1) + QUOTENAME(DB\_NAME()) AS [-- SQL STATEMENTS --],  1 AS [-- RESULT ORDER HOLDER --]  UNION  SELECT '' AS [-- SQL STATEMENTS --],  2 AS [-- RESULT ORDER HOLDER --]  UNION  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\* DB USER CREATION \*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  SELECT '-- [-- DB USERS --] --' AS [-- SQL STATEMENTS --],  3 AS [-- RESULT ORDER HOLDER --]  UNION  SELECT 'IF NOT EXISTS (SELECT [name] FROM sys.database\_principals WHERE [name] = ' + SPACE(1) + '''' + [name] + '''' + ') BEGIN CREATE USER ' + SPACE(1) + QUOTENAME([name]) + ' FOR LOGIN ' + QUOTENAME([name]) + ' WITH DEFAULT\_SCHEMA = ' + QUOTENAME([default\_schema\_name]) + SPACE(1) + 'END; ' AS [-- SQL STATEMENTS --],  4 AS [-- RESULT ORDER HOLDER --]  FROM sys.database\_principals AS rm  WHERE [type] IN ('U', 'S', 'G') -- windows users, sql users, windows groups  UNION  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\* DB ROLE PERMISSIONS \*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  SELECT '-- [-- DB ROLES --] --' AS [-- SQL STATEMENTS --],  5 AS [-- RESULT ORDER HOLDER --]  UNION  SELECT 'EXEC sp\_addrolemember @rolename ='  + SPACE(1) + QUOTENAME(USER\_NAME(rm.role\_principal\_id), '''') + ', @membername =' + SPACE(1) + QUOTENAME(USER\_NAME(rm.member\_principal\_id), '''') AS [-- SQL STATEMENTS --],  6 AS [-- RESULT ORDER HOLDER --]  FROM sys.database\_role\_members AS rm  WHERE USER\_NAME(rm.member\_principal\_id) IN (  --get user names on the database  SELECT [name]  FROM sys.database\_principals  WHERE [principal\_id] > 4 -- 0 to 4 are system users/schemas  and [type] IN ('G', 'S', 'U') -- S = SQL user, U = Windows user, G = Windows group  )  --ORDER BY rm.role\_principal\_id ASC  UNION  SELECT '' AS [-- SQL STATEMENTS --],  7 AS [-- RESULT ORDER HOLDER --]  UNION  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\* OBJECT LEVEL PERMISSIONS \*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  SELECT '-- [-- OBJECT LEVEL PERMISSIONS --] --' AS [-- SQL STATEMENTS --],  8 AS [-- RESULT ORDER HOLDER --]  UNION  SELECT CASE  WHEN perm.state <> 'W' THEN perm.state\_desc  ELSE 'GRANT'  END  + SPACE(1) + perm.permission\_name + SPACE(1) + 'ON ' + QUOTENAME(SCHEMA\_NAME(obj.schema\_id)) + '.' + QUOTENAME(obj.name) --select, execute, etc on specific objects  + CASE  WHEN cl.column\_id IS NULL THEN SPACE(0)  ELSE '(' + QUOTENAME(cl.name) + ')'  END  + SPACE(1) + 'TO' + SPACE(1) + QUOTENAME(USER\_NAME(usr.principal\_id)) COLLATE database\_default  + CASE  WHEN perm.state <> 'W' THEN SPACE(0)  ELSE SPACE(1) + 'WITH GRANT OPTION'  END  AS [-- SQL STATEMENTS --],  9 AS [-- RESULT ORDER HOLDER --]  FROM  sys.database\_permissions AS perm  INNER JOIN  sys.objects AS obj  ON perm.major\_id = obj.[object\_id]  INNER JOIN  sys.database\_principals AS usr  ON perm.grantee\_principal\_id = usr.principal\_id  LEFT JOIN  sys.columns AS cl  ON cl.column\_id = perm.minor\_id AND cl.[object\_id] = perm.major\_id  --WHERE usr.name = @OldUser  --ORDER BY perm.permission\_name ASC, perm.state\_desc ASC  UNION  SELECT '' AS [-- SQL STATEMENTS --],  10 AS [-- RESULT ORDER HOLDER --]  UNION  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\* DB LEVEL PERMISSIONS \*\*\*\*\*\*\*\*\*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  SELECT '-- [--DB LEVEL PERMISSIONS --] --' AS [-- SQL STATEMENTS --],  11 AS [-- RESULT ORDER HOLDER --]  UNION  SELECT CASE  WHEN perm.state <> 'W' THEN perm.state\_desc --W=Grant With Grant Option  ELSE 'GRANT'  END  + SPACE(1) + perm.permission\_name --CONNECT, etc  + SPACE(1) + 'TO' + SPACE(1) + '[' + USER\_NAME(usr.principal\_id) + ']' COLLATE database\_default --TO <user name>  + CASE  WHEN perm.state <> 'W' THEN SPACE(0)  ELSE SPACE(1) + 'WITH GRANT OPTION'  END  AS [-- SQL STATEMENTS --],  12 AS [-- RESULT ORDER HOLDER --]  FROM sys.database\_permissions AS perm  INNER JOIN  sys.database\_principals AS usr  ON perm.grantee\_principal\_id = usr.principal\_id  --WHERE usr.name = @OldUser  WHERE [perm].[major\_id] = 0  AND [usr].[principal\_id] > 4 -- 0 to 4 are system users/schemas  AND [usr].[type] IN ('G', 'S', 'U') -- S = SQL user, U = Windows user, G = Windows group  UNION  SELECT '' AS [-- SQL STATEMENTS --],  13 AS [-- RESULT ORDER HOLDER --]  UNION  SELECT '-- [--DB LEVEL SCHEMA PERMISSIONS --] --' AS [-- SQL STATEMENTS --],  14 AS [-- RESULT ORDER HOLDER --]  UNION  SELECT CASE  WHEN perm.state <> 'W' THEN perm.state\_desc --W=Grant With Grant Option  ELSE 'GRANT'  END  + SPACE(1) + perm.permission\_name --CONNECT, etc  + SPACE(1) + 'ON' + SPACE(1) + class\_desc + '::' COLLATE database\_default --TO <user name>  + QUOTENAME(SCHEMA\_NAME(major\_id))  + SPACE(1) + 'TO' + SPACE(1) + QUOTENAME(USER\_NAME(grantee\_principal\_id)) COLLATE database\_default  + CASE  WHEN perm.state <> 'W' THEN SPACE(0)  ELSE SPACE(1) + 'WITH GRANT OPTION'  END  AS [-- SQL STATEMENTS --],  15 AS [-- RESULT ORDER HOLDER --]  from sys.database\_permissions AS perm  inner join sys.schemas s  on perm.major\_id = s.schema\_id  inner join sys.database\_principals dbprin  on perm.grantee\_principal\_id = dbprin.principal\_id  WHERE class = 3 --class 3 = schema  ORDER BY [-- RESULT ORDER HOLDER --]  OPEN tmp  FETCH NEXT FROM tmp INTO @sql, @sort  WHILE @@FETCH\_STATUS = 0  BEGIN  PRINT @sql  FETCH NEXT FROM tmp INTO @sql, @sort  END  CLOSE tmp  DEALLOCATE tmp |

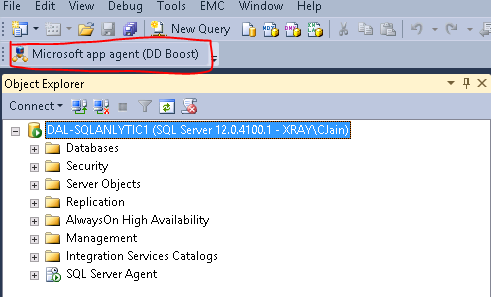
* Paste the above script in SSMS then change the DBName with required database name at the top in the script.
* Select result to file in SSMS to directly export the user permissions to SQL file as shown below.



* Then execute the script, it will ask for the location to save the user permissions file. Select the required location and save the file with .sql extension.

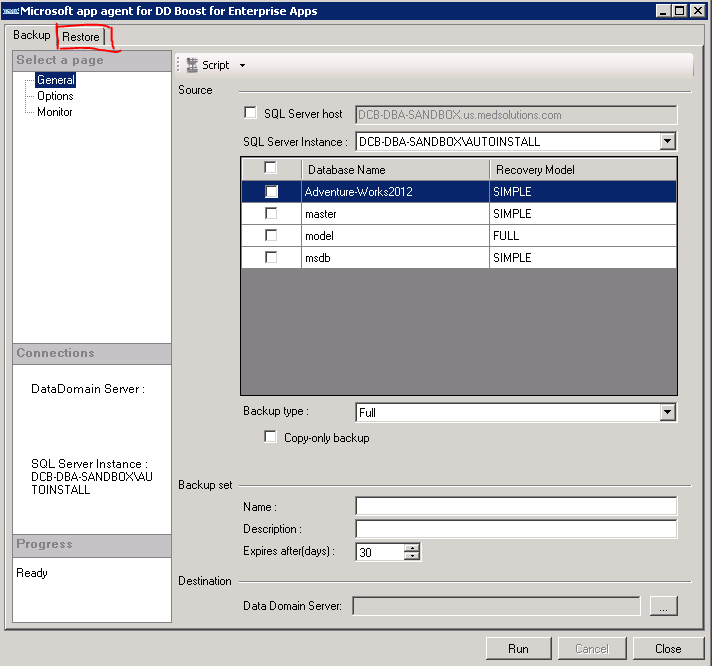


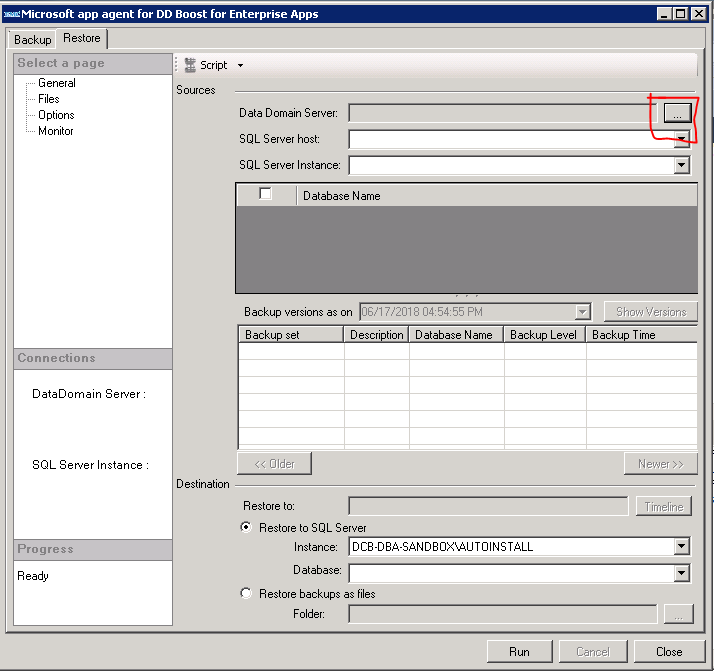
* Click on save. Repeat the same process for all the required databases to take user level permissions for all the required databases.
* If dababase refresh is from PROD to DEV, ask storage team to move the Prod copy from DAL location to ATL location.
* Add storage unit if it is not added to DDBoost.
* Open DDBoost as below.



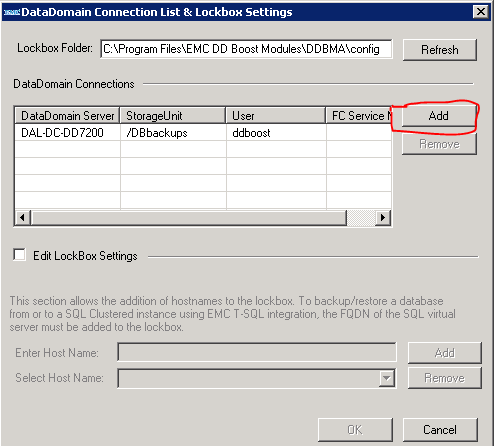
* If Microsoft app agent is not available in SSMS then open the DD Boost from the below path.
* C:\Program Files\DPSAPPS\MSAPPAGENT\bin\
* Select the file DDSSMSAddinApp then open RUN as Administrator.

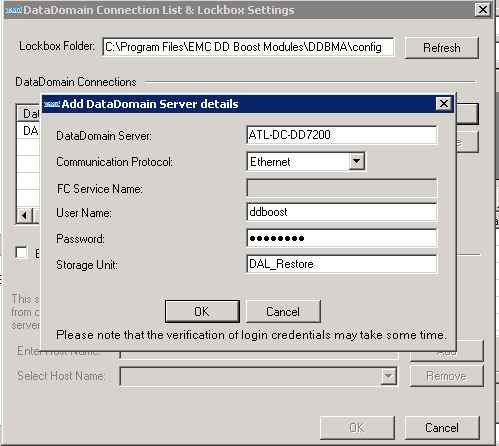
Click on Restore.





* Check if storage unit is not added please click on Add





DataDomain Server: DAL-DC-DD7200

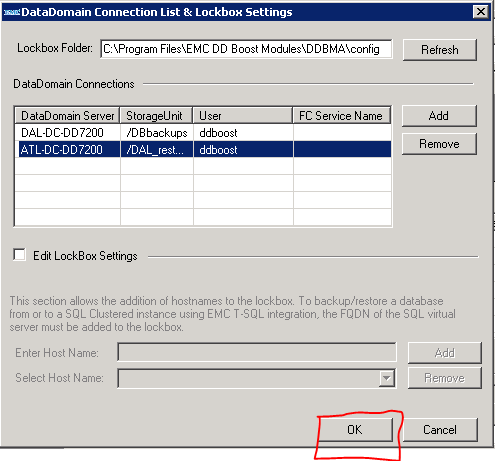
Communication Protocol: Ethernet

Username: ddboost

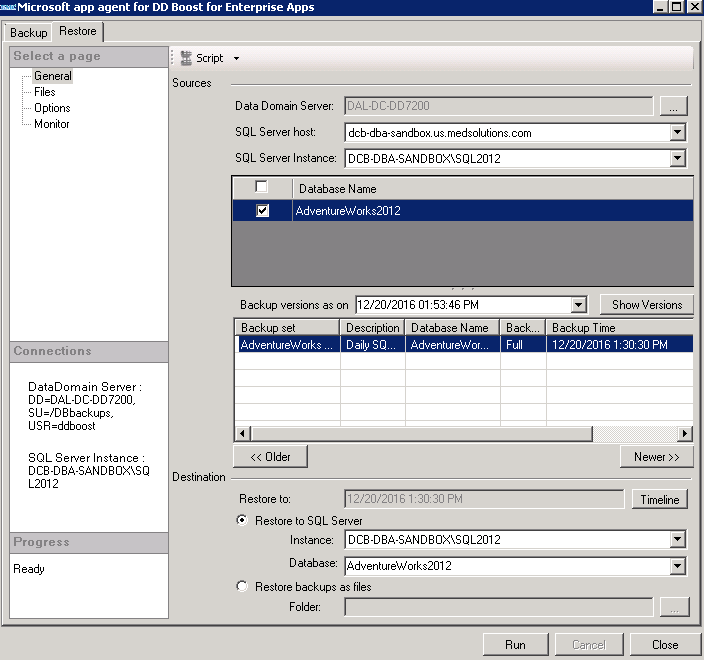
Pass: VMwar3!!

Storage unit: DBbackups

**These settings are case-sensitive so please make a note of upper and lower case used.**



* Once Storage unit is added to the DD Boost, select the requuired storage unit and click ok.
* In the DataDomain Connection List & Lockbox Settings, select the datadomain server and click OK.
* Sometimes, it may take some time at this step, so we just need to wait for a minute or two.
* Select the appropriate input values like follows:

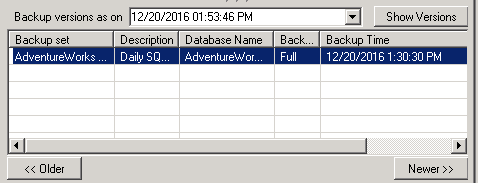


SQL Server is the host from where the backup was taken

Select the source SQL Server instance from where the backup was taken

Select the checkbox next to the database that you wish to restore.

The available backup set information will be populated in the table below



Select the backup. In this example, there is only one backup so only one record is visible.

In the Destination section at the bottom, there are two options:

1. Restore to SQL Server: select Instance where we need to restore to

Select the database to restore to (specify a new name if you want to create a new database for restoration) else you can choose to overwrite an existing database.

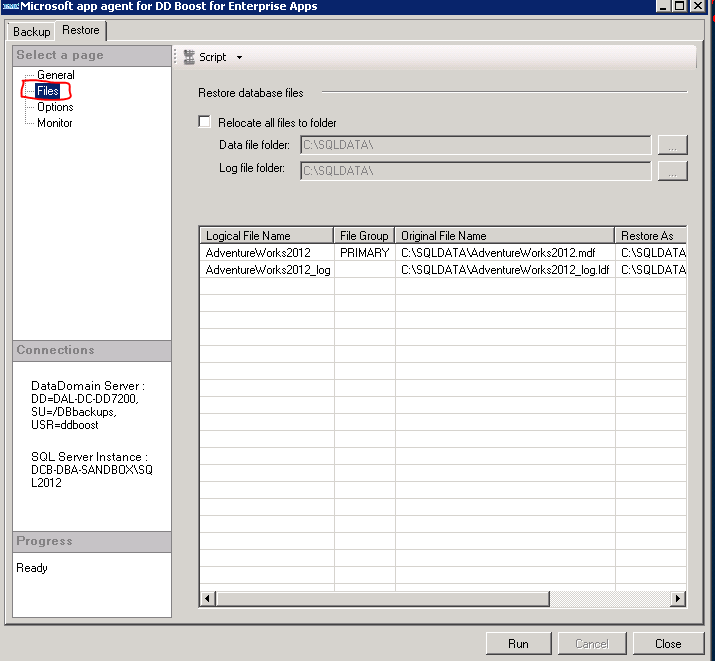
1. Restore backups as files: This will restore the backup as a native format file in uncompressed state.

If you have selected option 1, you need to Files tab and specify appropriate data/log file locations.

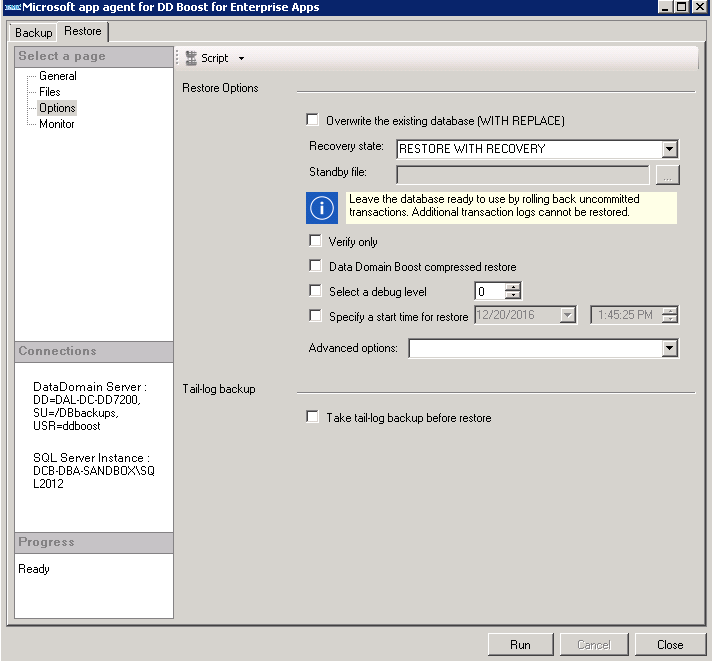
Execute the below script to find the physical location of files.

|  |
| --- |
| use DBName  go  sp\_helpfile |

From that above query find the logical and physical names change them at restore as option as below.



* Goto Options Tab and make appropriate selections like “restore with recovery” or “norecovery” or “standby”
* In the advanced options select keep replication or keep\_cdc (if applicable)
* In most cases, we will keep Take tail log backup restore as unchecked.
* Click Run or choose to generate a script which can be used for later. There are two different options like every other script:
* T-SQL Script: T-SQL script can be run from SSMS or SQL Agent Job Step
* CLI Script: Can be run from cmd or SQL Agent Job Step



* Before execute the restore script, kill all the sessions on required database because, if database is in use then restoration will be failed.
* Execute the below query to kill the sessions

|  |
| --- |
| USE [master];  DECLARE @kill varchar(8000) = '';  SELECT @kill = @kill + 'kill ' + CONVERT(varchar(5), session\_id) + ';'  FROM sys.dm\_exec\_sessions  WHERE database\_id = db\_id('MyDB')  EXEC(@kill); |

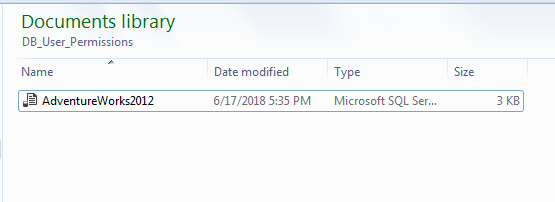
* Replace MyDB with required database name.
* Take the Tsql script from DD Boost and execute from SSMS. Execute the below query to find the status of database restoration.

|  |
| --- |
| select percent\_complete,\* from sys.dm\_exec\_requests where command like '%restore%' |

* Once restoration completed, execute the below query to get the script to delete all users on database.

|  |
| --- |
| declare @sql varchar(200),@name varchar(20)  declare cur cursor for  select name from sys.sysusers where name not in ('public','dbo','guest','INFORMATION\_SCHEMA','sys') and name not like '%db\_%'  open cur  fetch next from cur into @name  while @@FETCH\_STATUS = 0  begin  set @sql = 'drop user '+ @name + ' ;'  print @sql  --exec(@sql)  fetch next from cur into @name  end  close cur  deallocate cur |

* Copy output script and execute on the restored database. (make sure you are deleting users on required database)
* Once we have dropped source database users, execute the user level permission script which we have taken before DB refresh.



Change the recovery model and compatibility\_level if required.