Before you get started the script below check the following:

1. The restored database will have the same name as the backed up database
2. The restored database will be restored in the same location as the backed up database
3. The files have the following naming format
   * dbName\_YYYYMMDDHHMM.xxx
4. File extensions are as follows
   * Full backup – BAK
   * Differential backup – DIF
   * Transaction log backup – TRN
5. There should no missing transaction logs that may break the restore chain

So let's say we are creating our backups on the following schedule:

* Full backups at 12.00
* Differential backups every 3 hours starting at 12:10am
* Log backups every 15 minutes starting at 12.15am

The script below will search through the directory and create the restore script for us. The only two parameters that would need to change are the **@dbName** and the **@backupPath** for any other database

|  |
| --- |
| USE Master;  GO  SET NOCOUNT ON   -- 1 - Variable declaration  DECLARE @dbName sysname  DECLARE @backupPath NVARCHAR(500)  DECLARE @cmd NVARCHAR(500)  DECLARE @fileList TABLE (backupFile NVARCHAR(255))  DECLARE @lastFullBackup NVARCHAR(500)  DECLARE @lastDiffBackup NVARCHAR(500)  DECLARE @backupFile NVARCHAR(500)   -- 2 - Initialize variables  SET @dbName = 'ADMS'  SET @backupPath = ‘E:\22\_DB\ADMS\'   -- 3 - get list of files  SET @cmd = 'DIR /b ' + @backupPath   INSERT INTO @fileList(backupFile)  EXEC master.sys.xp\_cmdshell @cmd   -- 4 - Find latest full backup  SELECT @lastFullBackup = MAX(backupFile)  FROM @fileList  WHERE backupFile LIKE '%.BAK'  AND backupFile LIKE @dbName + '%'   SET @cmd = 'RESTORE DATABASE ' + @dbName + ' FROM DISK = '''  + @backupPath + @lastFullBackup + ''' WITH NORECOVERY, REPLACE'  PRINT @cmd   -- 4 - Find latest diff backup  SELECT @lastDiffBackup = MAX(backupFile)  FROM @fileList  WHERE backupFile LIKE '%.DIF'  AND backupFile LIKE @dbName + '%'  AND backupFile > @lastFullBackup   -- check to make sure there is a diff backup  IF @lastDiffBackup IS NOT NULL  BEGIN  SET @cmd = 'RESTORE DATABASE ' + @dbName + ' FROM DISK = '''  + @backupPath + @lastDiffBackup + ''' WITH NORECOVERY'  PRINT @cmd  SET @lastFullBackup = @lastDiffBackup  END   -- 5 - check for log backups  DECLARE backupFiles CURSOR FOR  SELECT backupFile  FROM @fileList  WHERE backupFile LIKE '%.TRN'  AND backupFile LIKE @dbName + '%'  AND backupFile > @lastFullBackup   OPEN backupFiles   -- Loop through all the files for the database  FETCH NEXT FROM backupFiles INTO @backupFile   WHILE @@FETCH\_STATUS = 0  BEGIN  SET @cmd = 'RESTORE LOG ' + @dbName + ' FROM DISK = '''  + @backupPath + @backupFile + ''' WITH NORECOVERY'  PRINT @cmd  FETCH NEXT FROM backupFiles INTO @backupFile  END   CLOSE backupFiles  DEALLOCATE backupFiles   -- 6 - put database in a useable state  SET @cmd = 'RESTORE DATABASE ' + @dbName + ' WITH RECOVERY'  PRINT @cmd |

If you execute the above search code in a query window, it will find out the latest full backup, differential backup and transactional backup in the output. At this point you can copy the code from the results page and paste the code into another query window and run the query to do the actual restore.

Note : once the restore is completed you need to refresh the database once to affect the changes. Below is the sample output results which we tried in the testing environment.

RESTORE DATABASE Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_000001\_3045910.bak' WITH NORECOVERY, REPLACE

RESTORE DATABASE Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_000001\_3172868.dif' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_001501\_4231386.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_003000\_7672424.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_004500\_8184046.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_010000\_8724966.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_011501\_0027634.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_013000\_8839972.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_014500\_9820362.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_020001\_0400346.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_021501\_0609222.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_023001\_1570080.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_024501\_1163698.trn' WITH NORECOVERY

RESTORE LOG Trinity FROM DISK = 'C:\sree\Trinity\_backup\_2014\_02\_11\_030001\_2251514.trn' WITH NORECOVERY

RESTORE DATABASE Trinity WITH RECOVERY

As you can see it does a Full restore, the latest Differential restore and all Transaction Logs after that. The script also does a WITH RECOVERY at the end to put the database in a useable state.

RESTORE VERIFYONLY FROM DISK = 'C:\sreetest\MtCarmel\_backup\_2014\_12\_21\_031000\_6342502.dif'

To check the history of the backups on the database. Only change the field of database\_name='Trinity'

SELECT

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 'Database'

WHEN 'L' THEN 'Log'

WHEN 'I' THEN 'Differential'

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 msdb.dbo.backupset.database\_name='Trinity' and (CONVERT(datetime, msdb.dbo.backupset.backup\_start\_date,

102) >=

GETDATE()

- 8)

ORDER BY

msdb.dbo.backupset.database\_name,

msdb.dbo.backupset.backup\_finish\_date desc

SP for restorescript

USE [master]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[ADMS] Script Date: 02/26/2015 08:41:40 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[ADMS]

-- Add the parameters for the stored procedure here

@BasePath NVARCHAR(2000),

@dbName NVARCHAR(500)

AS

BEGIN

DECLARE

--@BasePath varchar(1000)

@Path varchar(1000)

,@FullPath varchar(2000)

,@Id int;

-- 1 - Variable declaration

--DECLARE @dbName sysname

DECLARE @lastFullBackup NVARCHAR(4000)

DECLARE @lastDiffBackup NVARCHAR(4000)

DECLARE @backupFile NVARCHAR(4000)

----This is your starting point.

--SET @BasePath = 'D:\Backup';

--SET @dbName = 'BackupFiles'

--Create a temp table to hold the results.

IF OBJECT\_ID('tempdb..#ADMS\_DirectoryTree')IS NOT NULL

DROP TABLE #ADMS\_DirectoryTree;

CREATE TABLE #ADMS\_DirectoryTree (

id int IDENTITY(1,1)

,fullpath varchar(2000)

,subdirectory nvarchar(512)

,depth int

,isfile bit,

fileFullPath varchar(4000));

--Create a clustered index to keep everything in order.

ALTER TABLE #ADMS\_DirectoryTree ADD CONSTRAINT PK\_ADMS\_DirectoryTree PRIMARY KEY CLUSTERED (id);

--Populate the table using the initial base path.

INSERT #ADMS\_DirectoryTree (subdirectory,depth,isfile)

EXEC master.sys.xp\_dirtree @BasePath,1,1;

UPDATE #ADMS\_DirectoryTree SET fullpath = @BasePath;

--Loop through the table as long as there are still folders to process.

WHILE EXISTS (SELECT id FROM #ADMS\_DirectoryTree WHERE isfile = 0)

BEGIN

--Select the first row that is a folder.

SELECT TOP (1)

@Id = id

,@FullPath = fullpath

,@Path = @BasePath + '\' + subdirectory

FROM #ADMS\_DirectoryTree WHERE isfile = 0;

IF @FullPath = @Path

BEGIN

--Do this section if the we are still in the same folder.

INSERT #ADMS\_DirectoryTree (subdirectory,depth,isfile)

EXEC master.sys.xp\_dirtree @Path,1,1;

UPDATE #ADMS\_DirectoryTree

SET fullpath = @Path

WHERE fullpath IS NULL;

--Delete the processed folder.

DELETE FROM #ADMS\_DirectoryTree WHERE id = @Id;

END

ELSE

BEGIN

--Do this section if we need to jump down into another subfolder.

SET @BasePath = @FullPath;

--Select the first row that is a folder.

SELECT TOP (1)

@Id = id

,@FullPath = fullpath

,@Path = @BasePath + '\' + subdirectory

FROM #ADMS\_DirectoryTree WHERE isfile = 0;

INSERT #ADMS\_DirectoryTree (subdirectory,depth,isfile)

EXEC master.sys.xp\_dirtree @Path,1,1;

UPDATE #ADMS\_DirectoryTree

SET fullpath = @Path

WHERE fullpath IS NULL;

--Delete the processed folder.

DELETE FROM #ADMS\_DirectoryTree WHERE id = @Id;

END

END

--Output the results.

UPDATE #ADMS\_DirectoryTree SET fileFullPath=(fullpath + '\' + subdirectory)

SELECT @lastFullBackup=MAX(fileFullPath)

FROM #ADMS\_DirectoryTree

WHERE subdirectory LIKE '%.bak'

AND subdirectory LIKE @dbName + '%'

SET @backupFile = 'RESTORE DATABASE ' + @dbName + ' FROM DISK = '''

+ @lastFullBackup + ''' WITH NORECOVERY, REPLACE'

EXEC sys.sp\_executesql @backupFile

PRINT @backupFile

-- 4 - Find latest diff backup

SELECT @lastDiffBackup = MAX(fileFullPath)

FROM #ADMS\_DirectoryTree

WHERE subdirectory LIKE '%.DIF'

AND subdirectory LIKE @dbName + '%'

AND (fileFullPath) > @lastFullBackup

-- check to make sure there is a diff backup

IF @lastDiffBackup IS NOT NULL

BEGIN

SET @backupFile = 'RESTORE DATABASE ' + @dbName + ' FROM DISK = '''

+ @lastDiffBackup + ''' WITH NORECOVERY'

EXEC sys.sp\_executesql @backupFile

PRINT @backupFile

SET @lastFullBackup = @lastDiffBackup

END

select @lastFullBackup

SELECT (fileFullPath), @lastFullBackup

FROM #ADMS\_DirectoryTree

WHERE subdirectory LIKE '%.TRN'

AND subdirectory LIKE @dbName + '%'

AND fileFullPath > @lastFullBackup

-- 5 - check for log backups

DECLARE @trnFile NVARCHAR(500)

DECLARE backupFiles CURSOR FOR

SELECT (fileFullPath)

FROM #ADMS\_DirectoryTree

WHERE subdirectory LIKE '%.TRN'

AND subdirectory LIKE @dbName + '%'

AND fileFullPath > @lastFullBackup

OPEN backupFiles

-- Loop through all the files for the database

FETCH NEXT FROM backupFiles INTO @backupFile

WHILE @@FETCH\_STATUS = 0

BEGIN

SET @trnFile = 'RESTORE LOG ' + @dbName + ' FROM DISK = '''

+ @backupFile + ''' WITH NORECOVERY'

EXEC sys.sp\_executesql @trnFile

PRINT @trnFile

FETCH NEXT FROM backupFiles INTO @backupFile

END

CLOSE backupFiles

DEALLOCATE backupFiles

-- 6 - put database in a useable state

SET @trnFile = 'RESTORE DATABASE ' + @dbName + ' WITH RECOVERY'

EXEC sys.sp\_executesql @trnFile

PRINT @trnFile

--Cleanup.

IF OBJECT\_ID('tempdb..#ADMS\_DirectoryTree')IS NOT NULL

DROP TABLE #ADMS\_DirectoryTree;

END

GO

To Enable Xp\_cmdshell in SQL Server

-- To allow advanced options to be changed.

EXEC sp\_configure 'show advanced options', 1

GO

-- To update the currently configured value for advanced options.

RECONFIGURE

GO

-- To enable the feature.

EXEC sp\_configure 'xp\_cmdshell', 1

GO

-- To update the currently configured value for this feature.

RECONFIGURE

GO

USE [ADMS]

GO

CREATE USER [TrinityReportUser] FOR LOGIN [TrinityReportUser]

GO

USE [ADMS]

GO

EXEC sp\_addrolemember N'db\_datareader', N'TrinityReportUser'

GO

USE [ADMS]

GO

EXEC sp\_addrolemember N'db\_owner', N'TrinityReportUser'

GO

To create a job

exec ReportServer 'C:\Backup\ReportServer', 'ReportServer'

http://www.sqlservercentral.com/blogs/sqlinthewild/2011/03/08/full-backups\_2C00\_-the-log-chain-and-the-copy\_5F00\_only-option\_2E00\_/

https://social.msdn.microsoft.com/Forums/sqlserver/en-US/22740494-d65f-4f30-93e9-2c0c9d22a500/restoring-separate-differential-backup-files

how to check the LSN numbers

SELECT TOP 1 b.type, b.first\_lsn, b.last\_lsn, b.checkpoint\_lsn, b.database\_backup\_lsn  
FROM msdb..restorehistory a  
  
INNER JOIN msdb..backupset b ON a.backup\_set\_id = b.backup\_set\_id  
WHERE a.destination\_database\_name = 'AdventureWorks'  
  
ORDER BY restore\_date DESC

http://www.mssqltips.com/sqlservertip/1584/auto-generate-sql-server-restore-script-from-backup-files-in-a-directory/