Database Toolset

Toolset of tools and script for maintaining the databases. We also included scripts for different applications.

- ESP - Delete Old Messages
- ESP - Database Copy Script
- ESP - Delete User
- ESP - Create Financial Year and Period
- MsSQL - Create Database Table from Code
- ESP - Add Notifications to All Active Companies

- ESP - Delete Product Design in Imported Status
- ESP - Delete Old Message
- SQL - Kill Database Connections
- MsSQL - Backups
- MsSQL - Verify Properties
- MySQL - Bag Of Tricks
- MsSQL - SQL Reporting Services
- MsSQL - Move TempDB To Another Drive
- MsSQL - Set All User Tables to Simple on DEV/TEST Server

ESP - Delete Old Messages

```
/*

This will delete all old XML messages
and events older then 5 days

SFL 2008-11-29 Created

Status
1□FAILED
3□Sucessfull
6□History

*/

delete from infbigtext where foreignId <> 1 and lastchanged < dateadd(d,-5,getdate()) and foreigntype like
'Message%'
delete from infapplicationevent where createddate < dateadd(d,-5,getdate())

--Delete Messages too
declare @id int
declare look cursor for

Select id from mscmessage where processeddate < dateadd(d,-5,getdate()) and status in (3,6)

open look
fetch next from look into @id
while @@fetch_status=0
  begin
□update mscmessage set previousMessageID=null where previousMessageID=@id
□delete from mscmessage where id=@id

fetch next from look into @id
  end
close look
deallocate look
```

```
delete from infexception where mainttime < dateadd(dd,-5,getdate())
```

ESP - Database Copy Script

```
/*
Search and replace for the ESP database and its folders

Created By: SFL 08/13/23
Modified By: SFL 08/21/24

Use at own risk

*/
USE ESP_DEV;
/*
select * from infParameter where value like '%/%'
select * from infPlantValue where value like '%/%'
select * from infGroupValue where value like '%/%'
select * from infpersonalValue where value like '%/%'

select * from infParameter where value like '%\%'
select * from infPlantValue where value like '%\%'
select * from infGroupValue where value like '%\%'
select * from infpersonalValue where value like '%\%'
*/
--update infParameter set mainttime=getdate(),userid='CopyDB', value=replace(value,'\PRD\','\DEV\') where value like '%\PRD\%'

update infParameter set mainttime=getdate(),userid='CopyDB',
value=replace(value,'\\SFL-APP-001.onling.com\\KiwiplanInterfaces\Prd\','\\SFL-APP-001.onling.com\KiwiplanInterfaces\Dev\')
where value like '%\\SFL-APP-001.onling.com\%'

update infmapdataset set userid='CopyDB', mainttime=getdate(),hostuser='remuser',
hostpassword='l5oPU0kDgwy7vPj/uUFuqA==' where hostdataset ='fgsc'

update infParameter set mainttime=getdate(),userid='CopyDB', value='SFL-APP-001' where name ='Purge server name'
```

```
delete from infMAPDataset where id not in (select mapdatasetID from orgplant)
update infMAPDataset set mainttime=getdate(),userid='CopyDB', hostname='USW2-D-KP-MAP' where
hostdataset ='fgsc'

update infParameter set mainttime=getdate(), userid='CopyDB', value='TEST 9.80.5040' where
name='database status'

delete from agtAgentServer where id not in (select agentserverID from agtagent)
update agtAgentServer set mainttime=getdate(),userid='CopyDB', name='SFL-APP-001'

update agtagent set mainttime=getdate(),userid='CopyDB', agentServerID =(select id from agtAgentServer
where name='SFL-APP-001')

update infParameter set mainttime=getdate(),userid='CopyDB', value='SFL-APP-001' where name = 'KDG
server name'

/*
delete from infUserMembership where memberID > (select min(id) from infUser)
delete from infTaskbutton where usernameID not in (select memberid from infUserMembership )
delete from infUser where id > (select min(id) from infUser)
*/

--Need SID search and replace for all users tool ran

--create folders that do not exist after replace To DO!

update infparameter set userid='Move_to_', mainttime=getdate() , value=0 where name='Age in Days for
archiving' and value != 0
update orgContact set email = ''
update orgAddress set notificationemail = ''
```

ESP - Delete User

```
/*

Delete users from ESP which do not exist in Active Directory
Created By: Steve Ling 2024/03/07

The user name is the name from the SAM account in ADD

Or if you do the following select

select name from infuser

*/

Declare @username nvarchar(30), @UID int, @UPD int

Set @username = 'Steve.Ling'
Set @UPD = 0 --1 to delete 0 to select

--- No Changes below here

Set @UID = (select ID from infuser where name = @username)

if @UPD <> 1
Begin
select * from infTaskbutton where usernameID = @UID
select * from infUserMembership where memberID = @UID
select * from infuser where name = @username
End

if @UPD = 1
Begin
delete from infTaskbutton where usernameID = @UID
delete from infUserMembership where memberID = @UID
delete from infuser where name = @username
End
```

ESP - Create Financial Year and Period

```
/*
Script to auto add finacial year and its periods if they are missing
Created By: Steve Ling 2024/03/02

You may have to change the stating year depending
on your finacial year and periods

*/

GO

Declare @y nvarchar(4), @yid int, @ly nvarchar(4), @ny nvarchar(4)

Set @y = '2030'

Set @ny = (select @y+1)
print @ny

Set @ly = (Select CASE WHEN @ny & 3 = 0 AND (@ny % 25 <> 0 OR @ny & 15 = 0) THEN '0229' else '0228' end)
print @ly

INSERT INTO [dbo].[orgfinancialyear]
           ([mainttime]
           ,[userid]
           ,[name]
           ,[startdate]
           ,[enddate])
     VALUES
           (getdate()
           ,'Script'
           ,@y
```

```
        ,Convert(datetime,@y+'0601 00:00:00')
        ,Convert(datetime,@ny+'0530 23:59:59')
    )

set @yid = (select @@IDENTITY )

INSERT INTO [dbo].[orgfinancialperiod]
        ([mainttime]
        ,[userid]
        ,[endDate]
        ,[periodnumber]
        ,[name]
        ,[startdate]
        ,[financialyearID])
    VALUES
    (GETDATE(),'Script',Convert(datetime,@y+'0628 23:59:59'),1,'June',Convert(datetime,@y+'0601
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@y+'0802 23:59:59'),2,'July',Convert(datetime,@y+'0629
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@y+'0830 23:59:59'),3,'August',Convert(datetime,@y+'0803
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@y+'0927 23:59:59'),4,'September',Convert(datetime,@y+'0831
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@y+'1101 23:59:59'),5,'October',Convert(datetime,@y+'0928
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@y+'1129 23:59:59'),6,'November',Convert(datetime,@y+'1102
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@ny+'0103 23:59:59'),7,'December',Convert(datetime,@y+'1130
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@ny+'0131 23:59:59'),8,'January',Convert(datetime,@ny+'0104
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@ny+@ly+' 23:59:59'),9,'February',Convert(datetime,@ny+'0201
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@ny+'0328 23:59:59'),10,'March',Convert(datetime,@ny+'0301
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@ny+'0502 23:59:59'),11,'April',Convert(datetime,@ny+'0329
00:00:00'),@yid),
    (GETDATE(),'Script',Convert(datetime,@ny+'0530 23:59:59'),12,'May',Convert(datetime,@ny+'0503
00:00:00'),@yid)
```

```
select * from orgfinancialyear where id=@yid;select * from orgfinancialperiod where financialyearID =@yid
GO
```

```
/*
Script to auto add finacial year and its periods if they are missing
Created By: Steve Ling 2024/03/02

You may have to change the stating year depending
on your finacial year and periods

This one is January to December

*/

GO

Declare @y nvarchar(4), @yid int, @ly nvarchar(4)

Set @y = '2016'

Set @ly = (Select CASE WHEN @y & 3 = 0 AND (@y % 25 <> 0 OR @y & 15 = 0) THEN '0229' else '0228' end)

INSERT INTO [dbo].[orgfinancialyear]
           ([mainttime]
           ,[userid]
           ,[name]
           ,[startdate]
           ,[enddate])
     VALUES
           (getdate()
           ,'Script'
           ,@y
           ,Convert(datetime,@y+'0101 00:00:00')
           ,Convert(datetime,@y+'1231 23:59:59')
   )

set @yid = (select @@IDENTITY )

INSERT INTO [dbo].[orgfinancialperiod]
           ([mainttime]
```

```
        ,[userid]
        ,[endDate]
        ,[periodnumber]
        ,[name]
        ,[startdate]
        ,[financialyearID])
    VALUES
        (GETDATE(),'Script',Convert(datetime,@y+'0131 23:59:59'),1,'January',Convert(datetime,@y+'0101
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+@ly+' 23:59:59'),2,'February',Convert(datetime,@y+'0201
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'0331 23:59:59'),3,'March',Convert(datetime,@y+'0301
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'0430 23:59:59'),4,'April',Convert(datetime,@y+'0401
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'0531 23:59:59'),5,'May',Convert(datetime,@y+'0501
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'0630 23:59:59'),6,'June',Convert(datetime,@y+'0601
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'0731 23:59:59'),7,'July',Convert(datetime,@y+'0701
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'0831 23:59:59'),8,'August',Convert(datetime,@y+'0801
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'0930 23:59:59'),9,'September',Convert(datetime,@y+'0901
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'1031 23:59:59'),10,'October',Convert(datetime,@y+'1001
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'1130 23:59:59'),11,'November',Convert(datetime,@y+'1101
00:00:00'),@yid),
□□  (GETDATE(),'Script',Convert(datetime,@y+'1231 23:59:59'),12,'December',Convert(datetime,@y+'1201
00:00:00'),@yid)

--select * from orgfinancialyear where id=@yid;select * from orgfinancialperiod where financialyearID =@yid
GO
```

MsSQL - Create Database Table from Code

```
declare @TableName sysname = 'TableName'
declare @Result varchar(max) = 'public class ' + @TableName + '
{'

select @Result = @Result + '
    public ' + ColumnType + NullableSign + ' ' + ColumnName + ' { get; set; }
'
from
(
    select
        replace(col.name, ' ', '_') ColumnName,
        column_id ColumnId,
        case typ.name
            when 'bigint' then 'long'
            when 'binary' then 'byte[]'
            when 'bit' then 'bool'
            when 'char' then 'string'
            when 'date' then 'DateTime'
            when 'datetime' then 'DateTime'
            when 'datetime2' then 'DateTime'
            when 'datetimeoffset' then 'DateTimeOffset'
            when 'decimal' then 'decimal'
            when 'float' then 'double'
            when 'image' then 'byte[]'
            when 'int' then 'int'
            when 'money' then 'decimal'
            when 'nchar' then 'string'
            when 'ntext' then 'string'
            when 'numeric' then 'decimal'
            when 'nvarchar' then 'string'
            when 'real' then 'float'
            when 'smalldatetime' then 'DateTime'
```

```
            when 'smallint' then 'short'
            when 'smallmoney' then 'decimal'
            when 'text' then 'string'
            when 'time' then 'TimeSpan'
            when 'timestamp' then 'long'
            when 'tinyint' then 'byte'
            when 'uniqueidentifier' then 'Guid'
            when 'varbinary' then 'byte[]'
            when 'varchar' then 'string'
            else 'UNKNOWN_' + typ.name
        end ColumnType,
        case
            when col.is_nullable = 1 and typ.name in ('bigint', 'bit', 'date', 'datetime', 'datetime2', 'datetimeoffset',
'decimal', 'float', 'int', 'money', 'numeric', 'real', 'smalldatetime', 'smallint', 'smallmoney', 'time', 'tinyint',
'uniqueidentifier')
            then '?'
            else ''
        end NullableSign
    from sys.columns col
        join sys.types typ on
            col.system_type_id = typ.system_type_id AND col.user_type_id = typ.user_type_id
    where object_id = object_id(@TableName)
) t
order by ColumnId

set @Result = @Result  + '
}'

print @Result
```

Stored Procedure

ESP - Add Notifications to All Active Companies

```
/*
This script will add Notifications to all active companies
Please run in SQLAgent every 15 minutes.

Modified Steven F Ling 2019-11-27
Modified Steven F Ling 2023-08-14
*/
DECLARE @companyid  INT
DECLARE @name VARCHAR(100)
DECLARE @regionName VARCHAR(100)
DECLARE @not1id INT, @not2id INT, @not3id INT, @not4id INT, @not5id INT

--##### User Variables
--##### Please set these variables before running this script
DECLARE @updateUser VARCHAR(100) = 'AddNotificationScript'
DECLARE @messageSystemID1 INT = (SELECT id FROM mscmessagesystem WHERE name = 'Invoice_Export')
DECLARE @messageSystemID2 INT = (SELECT id FROM mscmessagesystem WHERE name = 'espOrderExport')
DECLARE @messageSystemID3 INT = (SELECT id FROM mscmessagesystem WHERE name = 'ESP Docket
Export')
DECLARE @messageSystemID4 INT = (SELECT id FROM mscmessagesystem WHERE name =
'CreditNote_Export')
DECLARE @messageSystemID5 INT = (SELECT id FROM mscmessagesystem WHERE name = 'espOrderExport')


-- Where clause added to look for only Active companies
DECLARE look CURSOR FOR SELECT c.id,c.name
      FROM orgcompany c
                  INNER JOIN orgaddress a ON c.billingAddressID=a.id
      LEFT Join orgNotification n on n.forCompanyID = c.id
                  WHERE companystatus='Active'
      and n.id is null
      --and n.deliverbyMessageSystemID != @messageSystemID3
```

```
                    ORDER BY c.id

□□□□□□

OPEN look
FETCH NEXT FROM look INTO @companyid, @name
WHILE @@FETCH_status=0
BEGIN

print ''
print '=====Starting with company and looking for Notifications: '+CONVERT(VARCHAR,@name)+'====='
print ''

--##### The notificationtype must be changed on each of the following lines to reflect the notifications being used
SELECT @not1id=(SELECT id FROM orgnotification WHERE forcompanyid=@companyid AND notificationtype ='InvoicePosted' and name ='Invoice')
SELECT @not2id=(SELECT id FROM orgnotification WHERE FORcompanyid=@companyid AND notificationtype ='OrderConfirmed')
SELECT @not3id=1 --(SELECT id FROM orgnotification WHERE FORcompanyid=@companyid AND notificationtype ='DocketImported')
SELECT @not4id=1 --(SELECT id FROM orgnotification WHERE forcompanyid=@companyid AND notificationtype ='InvoicePosted' and name ='Credit')
SELECT @not5id=(SELECT id FROM orgnotification WHERE FORcompanyid=@companyid AND notificationtype ='OrderCancelled')
--##### END notificationtype changes

prINT 'Notification adding process FOR: '+'Company ID: '+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name: '+CONVERT(VARCHAR,@name)

--Check to see if notification 1 exists the begin`
--IF @not1id IS NULL
--□□□□ BEGIN

--□□□□ print 'Adding Invoice Export FOR company: '+'Company ID: '+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name: '+CONVERT(VARCHAR,@name)

--□□□□□ INSERT INTO orgNotification (mainttime, userid, notificationtype, warninglevel, contenttype, contentname, forCompanyID, deliverbyMessageSystemID, toAddressID, toContactID, name, referenceRule)
--□□□□□
```

```
VALUES(GETDATE(),@updateUser,'InvoicePosted',0,2,'Invoice_Export',@companyid,@messageSystemID1,NULL,
NULL,'SAPInvoiceExport',null)

--□□□□ print 'Added'
--□□□□ END
--□□□□ Else
--□□□□ print 'Already exists: '+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name:
'+CONVERT(VARCHAR,@name)

/*   Comment out this section if you are not using Order Confirmations  */
 IF @not2id IS NULL
                BEGIN

                print 'Adding the ESP_Order Confirmation Export NotIFication FOR company: '+'Company ID:
'+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name: '+CONVERT(VARCHAR,@name)

□□□□□INSERT INTO orgNotification (mainttime, userid, notificationtype, warninglevel, contenttype, contentname,
forCompanyID, deliverbyMessageSystemID, toAddressID, toContactID, name, referenceRule)
□□□□□VALUES(GETDATE(),@updateUser,'OrderConfirmed',0,2,'ESPOrder_Export',@companyid,@messageSystem
ID2,NULL,NULL,'OrderConfirmed', null)

                print 'Added'
                END
□□□□ Else
□□□□ print 'Already exists: '+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name:
'+CONVERT(VARCHAR,@name)

/*  This is the Docket Export Section */
--IF @not3id IS NULL
--                BEGIN
--                print 'Added the ESP_Docket Export FOR company: '+'Company ID:
'+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name: '+CONVERT(VARCHAR,@name)

--□□□□□INSERT INTO orgNotification (mainttime, userid, notificationtype, warninglevel, contenttype, contentname,
forCompanyID, deliverbyMessageSystemID, toAddressID, toContactID, name, referenceRule)
--□□□□□VALUES(GETDATE(),@updateUser,'DocketFirstShipped',0,2,'ESP Docket
Export',@companyid,@messageSystemID3,NULL,NULL,'ESP_Docket Export Notification',
'[masterbilloflading.masterbolnumber]&"_"&[docketnumber]&"_"&[plant.plantcode]&"_"')

--                print 'Added'
```

```
--                END
--□□□□Else
--□□□□print 'Already exists: '+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name:
'+CONVERT(VARCHAR,@name)

--Check to see if notification 4 exists the begin`
--IF @not4id IS NULL
--□□□□ BEGIN

--□□□□ print 'Adding the Invoice Export FOR company: '+'Company ID: '+CONVERT(VARCHAR,@companyid)+ '
'+'Company Name: '+CONVERT(VARCHAR,@name)

--□□□□ INSERT INTO orgNotification (mainttime, userid, notificationtype, warninglevel, contenttype, contentname,
forCompanyID, deliverbyMessageSystemID, toAddressID, toContactID, name, referenceRule)
--□□□□
VALUES(GETDATE(),@updateUser,'InvoicePosted',0,2,'SFR015_CreditNote_Export',@companyid,@messageSyste
mID4,NULL,NULL,'SAPCreditExport',null)

--□□□□ print 'Added'
--□□□□ END
--□□□□ Else
--□□□□ print 'Already exists: '+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name:
'+CONVERT(VARCHAR,@name)

--Check to see if notification 5 exists the begin`
IF @not5id IS NULL
                BEGIN

                print 'Adding the ESP_Order Cancelled Export NotIfication FOR company: '+'Company ID:
'+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name: '+CONVERT(VARCHAR,@name)

□□□□□INSERT INTO orgNotification (mainttime, userid, notificationtype, warninglevel, contenttype, contentname,
forCompanyID, deliverbyMessageSystemID, toAddressID, toContactID, name, referenceRule)
□□□□□VALUES(GETDATE(),@updateUser,'OrderCancelled',0,2,'ESPOrder_Export',@companyid,@messageSystemI
D2,NULL,NULL,'OrderCancelled', null)

                print 'Added'
                END
□□□□ Else
□□□□ print 'Already exists: '+CONVERT(VARCHAR,@companyid)+ ' '+'Company Name:
```

```
'+CONVERT(VARCHAR,@name)

FETCH NEXT FROM look INTO @companyid,@name
print ''
print '=====Ended with '+'Company Name: '+CONVERT(VARCHAR,@name)+' ====='
print ''
print ''
END

CLOSE look
DEALLOCATE look
```

ESP - Delete Product Design in Imported Status

```
/***********************************************
This script is to delete PD that do not have
Orders on them and are in import status

20140207 Steven F Ling created

20230401 Steven F Ling Modified

Changed for the new Database structure

***********************************************/

declare @dnid int
declare @rid int
declare @dn varchar(30)
declare @slid int
declare @pdpid int
declare @udid int

declare look cursor for
SELECT isnull(pd.ID,0), isnull(pd.designnumber,''), isnull(rt.ID,0), isnull(sl.id,0), isnull(pdp.id,0), isnull(ud.id,0)
FROM ebxproductDesign pd with (NOLOCK)
LEFT JOIN ebxroute rt
ON pd.ID = rt.productDesignID
LEFT JOIN ebxProductDesignPlant pdp
ON pdp.productDesignID = pd.id
LEFT JOIN esporder o
ON o.productDesignID = pd.id
LEFT JOIN fgsstockline sl
ON sl.productdesignid = pd.id
LEFT JOIN ebxunitizingData ud
ON pdp.unitizingDataID = ud.id
```

```
inner join orgcompany c
on c.id = pd.companyID
where productdesignstatus = 'imported'
and o.productDesignID is null
--and designnumber in ('')
and c.companynumber ='1555'



open look
fetch next from look into @dnid, @dn ,@rid, @slid, @pdpid, @udid
while @@fetch_status=0
begin

IF @dnid > 0
Begin

print 'preping to delete '+@dn +' with ID of '+convert(nvarchar(30),@dnid)
print ' deleting ebxPriceListProductDesign'
delete from ebxPriceListProductDesign where productdesignid=@dnid
print ' deleting ebxproductprice'
delete from ebxproductprice where productdesignid=@dnid
print ' deleting cstcostestimateline'
delete from cstcostestimateline where productdesignid=@dnid
print ' deleting cstcostestimate'
delete from cstcostestimate where productdesignid=@dnid
print ' deleting ebxalternateRoute'
delete from ebxalternateRoute where productdesignid=@dnid
print ' deleting ebxmachinestep'
delete from ebxmachinestep where routeid in (@rid)
print ' deleting cstpurchaseCost PD'
delete from cstpurchaseCost where productDesignID = @dnid
print ' deleting cstpurchaseCost Route'
delete from cstpurchaseCost where routeID = @rid
print ' deleting ebxroute'
delete from ebxroute where productdesignid=@dnid
print ' deleting ebxProductDesignColourCoating'
delete from ebxProductDesignColourCoating where productDesignID = @dnid
print ' deleting ebxproductDesignInstruction'
delete from ebxproductDesignInstruction where productDesignID =  @dnid
print ' Updating ebxunitizingData'
```

update ebxunitizingData set ownerProductDesignPlantID=null where id=@udid
print 'deleting ebxProductDesignPlant'
delete from ebxProductDesignPlant where productDesignID = @dnid
print 'deleting ebxunitizingData'
delete from ebxunitizingData where id=@udid
print 'deleting fgsstocklinePurchaser'
delete from fgsstocklinePurchaser where stockLineID = @slid
print 'deleting fgsstockLine'
delete from fgsstockLine where productDesignID = @dnid
print 'updating ebxproductDesign'
Update ebxproductDesign set previousProductDesignID=null, nextProductDesignID=null where ID = @dnid
print 'updating ebxproductDesign'
Update ebxproductDesign set mainttime=getdate(), userid='Removed PD Link', previousProductDesignID=null
where previousProductDesignID = @dnid
print 'updating ebxproductDesign'
Update ebxproductDesign set mainttime=getdate(), userid='Removed PD Link', nextProductDesignID=null
where nextProductDesignID = @dnid
print 'deleting ebxproductDesign'
delete from ebxproductdesign where designnumber = @dn

End
Else
Begin
print 'does not exist...'
end

fetch next from look into @dnid, @dn ,@rid, @slid, @pdpid, @udid
end
close look
deallocate look

ESP - Delete Old Message

```
/*

This will delete all old XML messages
and events older then 5 days

SFL 2008-11-29 Created

Status
6 = history
5 =
4 = processing
3 = success
2 = pending
1 = failed
0 = created

*/

delete from infbigtext where foreignId <> 1 and lastchanged < dateadd(d,-5,getdate()) and foreigntype like
'Message%'
delete from infapplicationevent where createddate < dateadd(d,-5,getdate())

--Delete Messages too
declare @id int
declare look cursor for

Select id from mscmessage where processeddate < dateadd(d,-5,getdate()) and status in (3,6)

open look
fetch next from look into @id
while @@fetch_status=0
  begin
update mscmessage set previousMessageID=null where previousMessageID=@id
delete from mscmessage where id=@id
```

```
fetch next from look into @id
  end
close look
deallocate look
delete from infexception where mainttime < dateadd(dd,-5,getdate())
```

SQL - Kill Database Connections

```
/*

This is to kill off any connection to a database

Created by: Steve Ling 2022/03/20

*/

--This script will kill when ran

USE MASTER
GO

DECLARE @Spid INT
DECLARE @ExecSQL VARCHAR(255)

DECLARE KillCursor CURSOR LOCAL STATIC READ_ONLY FORWARD_ONLY
FOR
SELECT DISTINCT SPID
FROM MASTER..SysProcesses
WHERE DBID = DB_ID('ESP_TEST')

OPEN KillCursor

-- Grab the first SPID
FETCH NEXT
FROM KillCursor
INTO @Spid

WHILE @@FETCH_STATUS = 0
BEGIN
  SET @ExecSQL = 'KILL ' + CAST(@Spid AS VARCHAR(50))
```

```
□□EXEC□(@ExecSQL)

□□-- Pull the next SPID
        FETCH□NEXT
□□FROM□KillCursor
□□INTO□@Spid
□END

CLOSE□KillCursor

DEALLOCATE□KillCursor

-- this script will give you selects and also kill

USE□MASTER
GO

DECLARE @kill varchar(8000) = '';

SELECT @kill = @kill + 'kill ' + CONVERT(varchar(5), spid) + ';'
--select *
FROM master..sysprocesses

WHERE dbid = db_id('ESP_DEV')

EXEC(@kill);
```

MsSQL - Backups

Introduction

This is to create auto backups on a SQL server and place them on a drive.

Credits go to:

Ola Hallengren

https://ola.hallengren.com

The 20240104_Backup_ola_original.sql file is the untouched default version.

20240104_Backup_ola_original.sql

The 20240104_Create_Backups_Integrity_Index_Check.sql is the version used on ESP SQL servers with the use of a DB_Administration table.

20240104_Create_Backups_Integrity_Index_Check.sql

MsSQL - Verify Properties

```
/*

This will check the proprties on a given Database and also the server

Created By: Steve Ling 2022/04/07

*/

--Change the DB name here
DECLARE @DATABASE NVARCHAR(50) = 'SHOP'



/*
No changes below this line
*/
---- Simple Version Query
--SELECT    SERVERPROPERTY(   'productversion'   )   AS    "Product Version"   ,
--   SERVERPROPERTY (   'productlevel'   )   AS    "Product Level"   ,
--   SERVERPROPERTY (   'edition'   )   AS    "Edition"
-- Full Properties Query
 SELECT
 SERVERPROPERTY(   'ComputerNamePhysicalNetBIOS'   )   AS    'Current Failover Machine'   ,
 SERVERPROPERTY(   'MachineName'   )   AS    'Main Machine Name'   ,
 SERVERPROPERTY(   'ServerName'   )   AS    'Server\Instance Name'   ,
 SERVERPROPERTY(   'InstanceName'   )   AS    'Instance Name'   ,
 SERVERPROPERTY(   'ProcessID'   )   AS    'Instance ProcessID'   ,

 CASE    SUBSTRING   (   CONVERT   (VARCHAR(50), SERVERPROPERTY(   'productversion'   )), 1, 4)
    WHEN    '8.00'    THEN    'SQL2000'
     WHEN    '9.00'    THEN    'SQL2005'
     WHEN    '10.0'    THEN    'SQL2008'
     WHEN    '10.5'    THEN    'SQL2008 R2'
    WHEN    '11.0'    THEN    'SQL2012'
    WHEN    '12.0'    THEN    'SQL2014'
```

```
   ELSE   'Undetermined'   END   AS   'SQL Version'   ,

SERVERPROPERTY(   'ProductVersion'   )   AS   'Product Version'   ,
SERVERPROPERTY(   'ProductLevel'   )   AS   'Product Level'   ,
SERVERPROPERTY(   'Edition'   )   AS   'Edition'   ,

CASE   SERVERPROPERTY(   'EngineEdition'   )
   WHEN   1   THEN   'Person / Desktop'
   WHEN   2   THEN   'Standard'
   WHEN   3   THEN   'Enterprise'
   WHEN   4   THEN   'Express'
   WHEN   5   THEN   'SQL Database'
   ELSE   'Unknown'   END    AS   EngineEdition,



CASE   SERVERPROPERTY(   'IsIntegratedSecurityOnly'   )
   WHEN   1   THEN   'Window Authentication'
   WHEN   0   THEN   'Windows and SQL Authenication'
   ELSE   'Unknown'   END    AS   'Security Mode'   ,

SERVERPROPERTY(   'Collation'   )   AS   'Collation'   ,
SERVERPROPERTY(   'BuildClrVersion'   )   AS   'CLR Version'   ,

CASE   SERVERPROPERTY(   'IsFullTextInstalled'   )
   WHEN   0   THEN   'Installed'
   WHEN   1   THEN   'Not Installed'
   ELSE   'Unknown'   END   AS   'Full Text Indexing'   ,

CASE   SERVERPROPERTY(   'IsHadrEnabled'   )
   WHEN   0   THEN   'Disabled'
   WHEN   1   THEN   'Enabled'
   ELSE   'Unknown'   END    AS   'AlwaysOn Availability Groups'   ,

CASE   SERVERPROPERTY(   'HadrManagerStatus'   )
   WHEN   0   THEN   'Not Started / PENDING'
   WHEN   1   THEN   'Start/Running'
   WHEN   2   THEN   'Not Started / FAILED'
   ELSE   'Unknown'   END   AS   'AlwaysOn Availability Groups Manager'   ,

CASE   SERVERPROPERTY(   'IsClustered'   )
```

```
    WHEN   0   THEN    'Clustered'
    WHEN   1   THEN    'Not Clustered'
    ELSE    'Unknown'   END   AS    'Failover Cluster'   ,

 CASE   SERVERPROPERTY(   'IsSingleUser'   )
    WHEN   0   THEN    'Not in Single User Mode'
    WHEN   1   THEN    'In Single User Mode'
    ELSE    'Unknown'   END   AS    'Single User Mode'   ;

DECLARE @DB SYSNAME = @DATABASE

SELECT 'Collation' as Property, DATABASEPROPERTYEX (@DB, 'Collation') as Value UNION
SELECT 'ComparisonStyle' as Property, DATABASEPROPERTYEX (@DB, 'ComparisonStyle') as Value UNION
SELECT 'Edition' as Property, DATABASEPROPERTYEX (@DB, 'Edition') as Value UNION
SELECT 'IsAnsiNullDefault' as Property, DATABASEPROPERTYEX (@DB, 'IsAnsiNullDefault') as Value UNION
SELECT 'IsAnsiNullsEnabled' as Property, DATABASEPROPERTYEX (@DB, 'IsAnsiNullsEnabled') as Value UNION
SELECT 'IsAnsiPaddingEnabled' as Property, DATABASEPROPERTYEX (@DB, 'IsAnsiPaddingEnabled') as Value
UNION
SELECT 'IsAnsiWarningsEnabled' as Property, DATABASEPROPERTYEX (@DB, 'IsAnsiWarningsEnabled') as Value
UNION
SELECT 'IsArithmeticAbortEnabled' as Property, DATABASEPROPERTYEX (@DB, 'IsArithmeticAbortEnabled') as
Value UNION
SELECT 'IsAutoClose' as Property, DATABASEPROPERTYEX (@DB, 'IsAutoClose') as Value UNION
SELECT 'IsAutoCreateStatistics' as Property, DATABASEPROPERTYEX (@DB, 'IsAutoCreateStatistics') as Value
UNION
SELECT 'IsAutoCreateStatisticsIncremental' as Property, DATABASEPROPERTYEX (@DB,
'IsAutoCreateStatisticsIncremental') as Value UNION
SELECT 'IsAutoShrink' as Property, DATABASEPROPERTYEX (@DB, 'IsAutoShrink') as Value UNION
SELECT 'IsAutoUpdateStatistics' as Property, DATABASEPROPERTYEX (@DB, 'IsAutoUpdateStatistics') as Value
UNION
SELECT 'IsClone' as Property, DATABASEPROPERTYEX (@DB, 'IsClone') as Value UNION
SELECT 'IsCloseCursorsOnCommitEnabled' as Property, DATABASEPROPERTYEX (@DB,
'IsCloseCursorsOnCommitEnabled') as Value UNION
SELECT 'IsFulltextEnabled' as Property, DATABASEPROPERTYEX (@DB, 'IsFulltextEnabled') as Value UNION
SELECT 'IsInStandBy' as Property, DATABASEPROPERTYEX (@DB, 'IsInStandBy') as Value UNION
SELECT 'IsLocalCursorsDefault' as Property, DATABASEPROPERTYEX (@DB, 'IsLocalCursorsDefault') as Value
UNION
SELECT 'IsMemoryOptimizedElevateToSnapshotEnabled' as Property, DATABASEPROPERTYEX (@DB,
'IsMemoryOptimizedElevateToSnapshotEnabled') as Value UNION
SELECT 'IsMergePublished' as Property, DATABASEPROPERTYEX (@DB, 'IsMergePublished') as Value UNION
```

```
SELECT 'IsNullConcat' as Property, DATABASEPROPERTYEX (@DB, 'IsNullConcat') as Value UNION
SELECT 'IsNumericRoundAbortEnabled' as Property, DATABASEPROPERTYEX (@DB,
'IsNumericRoundAbortEnabled') as Value UNION
SELECT 'IsParameterizationForced' as Property, DATABASEPROPERTYEX (@DB, 'IsParameterizationForced') as
Value UNION
SELECT 'IsQuotedIdentifiersEnabled' as Property, DATABASEPROPERTYEX (@DB, 'IsQuotedIdentifiersEnabled') as
Value UNION
SELECT 'IsPublished' as Property, DATABASEPROPERTYEX (@DB, 'IsPublished') as Value UNION
SELECT 'IsRecursiveTriggersEnabled' as Property, DATABASEPROPERTYEX (@DB, 'IsRecursiveTriggersEnabled')
as Value UNION
SELECT 'IsSubscribed' as Property, DATABASEPROPERTYEX (@DB, 'IsSubscribed') as Value UNION
SELECT 'IsSyncWithBackup' as Property, DATABASEPROPERTYEX (@DB, 'IsSyncWithBackup') as Value UNION
SELECT 'IsTornPageDetectionEnabled' as Property, DATABASEPROPERTYEX (@DB, 'IsTornPageDetectionEnabled')
as Value UNION
SELECT 'IsVerifiedClone' as Property, DATABASEPROPERTYEX (@DB, 'IsVerifiedClone') as Value UNION
SELECT 'IsXTPSupported' as Property, DATABASEPROPERTYEX (@DB, 'IsXTPSupported') as Value UNION
SELECT 'LastGoodCheckDbTime' as Property, DATABASEPROPERTYEX (@DB, 'LastGoodCheckDbTime') as Value
UNION
SELECT 'LCID' as Property, DATABASEPROPERTYEX (@DB, 'LCID') as Value UNION
SELECT 'MaxSizeInBytes' as Property, DATABASEPROPERTYEX (@DB, 'MaxSizeInBytes') as Value UNION
SELECT 'Recovery' as Property, DATABASEPROPERTYEX (@DB, 'Recovery') as Value UNION
SELECT 'ServiceObjective' as Property, DATABASEPROPERTYEX (@DB, 'ServiceObjective') as Value UNION
SELECT 'ServiceObjectiveId' as Property, DATABASEPROPERTYEX (@DB, 'ServiceObjectiveId') as Value UNION
SELECT 'SQLSortOrder' as Property, DATABASEPROPERTYEX (@DB, 'SQLSortOrder') as Value UNION
SELECT 'Status' as Property, DATABASEPROPERTYEX (@DB, 'Status') as Value UNION
SELECT 'Updateability' as Property, DATABASEPROPERTYEX (@DB, 'Updateability') as Value UNION
SELECT 'UserAccess' as Property, DATABASEPROPERTYEX (@DB, 'UserAccess') as Value UNION
SELECT 'Version' as Property, DATABASEPROPERTYEX (@DB, 'Version') as Value

DECLARE @props TABLE (propertyname sysname PRIMARY KEY)
INSERT INTO @props(propertyname)
SELECT 'BuildClrVersion'
UNION
SELECT 'Collation'
UNION
SELECT 'CollationID'
UNION
SELECT 'ComparisonStyle'
UNION
SELECT 'ComputerNamePhysicalNetBIOS'
```

```
UNION
SELECT 'Edition'
UNION
SELECT 'EditionID'
UNION
SELECT 'EngineEdition'
UNION
SELECT 'InstanceName'
UNION
SELECT 'IsClustered'
UNION
SELECT 'IsFullTextInstalled'
UNION
SELECT 'IsIntegratedSecurityOnly'
UNION
SELECT 'IsSingleUser'
UNION
SELECT 'LCID'
UNION
SELECT 'LicenseType'
UNION
SELECT 'MachineName'
UNION
SELECT 'NumLicenses'
UNION
SELECT 'ProcessID'
UNION
SELECT 'ProductVersion'
UNION
SELECT 'ProductLevel'
UNION
SELECT 'ResourceLastUpdateDateTime'
UNION
SELECT 'ResourceVersion'
UNION
SELECT 'ServerName'
UNION
SELECT 'SqlCharSet'
UNION
SELECT 'SqlCharSetName'
```

```
UNION
SELECT 'SqlSortOrder'
UNION
SELECT 'SqlSortOrderName'
UNION
SELECT 'FilestreamShareName'
UNION
SELECT 'FilestreamConfiguredLevel'
UNION
SELECT 'FilestreamEffectiveLevel'

SELECT propertyname, SERVERPROPERTY(propertyname) FROM @props
```

MySQL - Bag Of Tricks

Introduction

This document has many useful command.

MySql Kill users

You can use the following syntax to lookup users within a database

```
SHOW PROCESSLIST;
SELECT group_concat(concat('KILL ',id,';') SEPARATOR ' \n') FROM information_schema.processlist WHERE `db` LIKE 'ashe%' AND `user` ='kiwilive';
```

Then cut and paste into a query window

MsSQL - SQL Reporting Services

Enable Errors

1. Connect to the database engine in SSMS and navigate to the ReportServer database.
2. Query the ConfigurationInfo table to get familiar with it. Issue the following query:

```
USE ReportServer
GO
UPDATE ConfigurationInfo SET Value = 'True' WHERE Name = 'EnableRemoteErrors'
```

3. Restart Reporting Services on the server: Click Start > Administrative Tools > Services to open the Services management console. Right-click the SQL Server Reporting Services ([InstanceName]) service, and then click Restart.

MsSQL - Move TempDB To Another Drive

Sometimes as a database administrator, you need to move the TempDB database and log files to a new hard drive.

This happens for example if you have installed SQL Server on C:\ and you no longer have space to process your queries. So it is necessary to move TempDB to a disk with more free space.

This article explains all the steps to move TempDB files.

Steps to move TempDB and log files to new location

Here are the steps to move SQL Server temporary database :

- Identify the location of TempDB data and log files
- Change the location of TempDB data and log files using ALTER DATABASE
- Stop and restart the SQL Server service
- Check path change
- Delete old TempDB as well as .mdf and .ldf files

Identify the location of TempDB data and log files

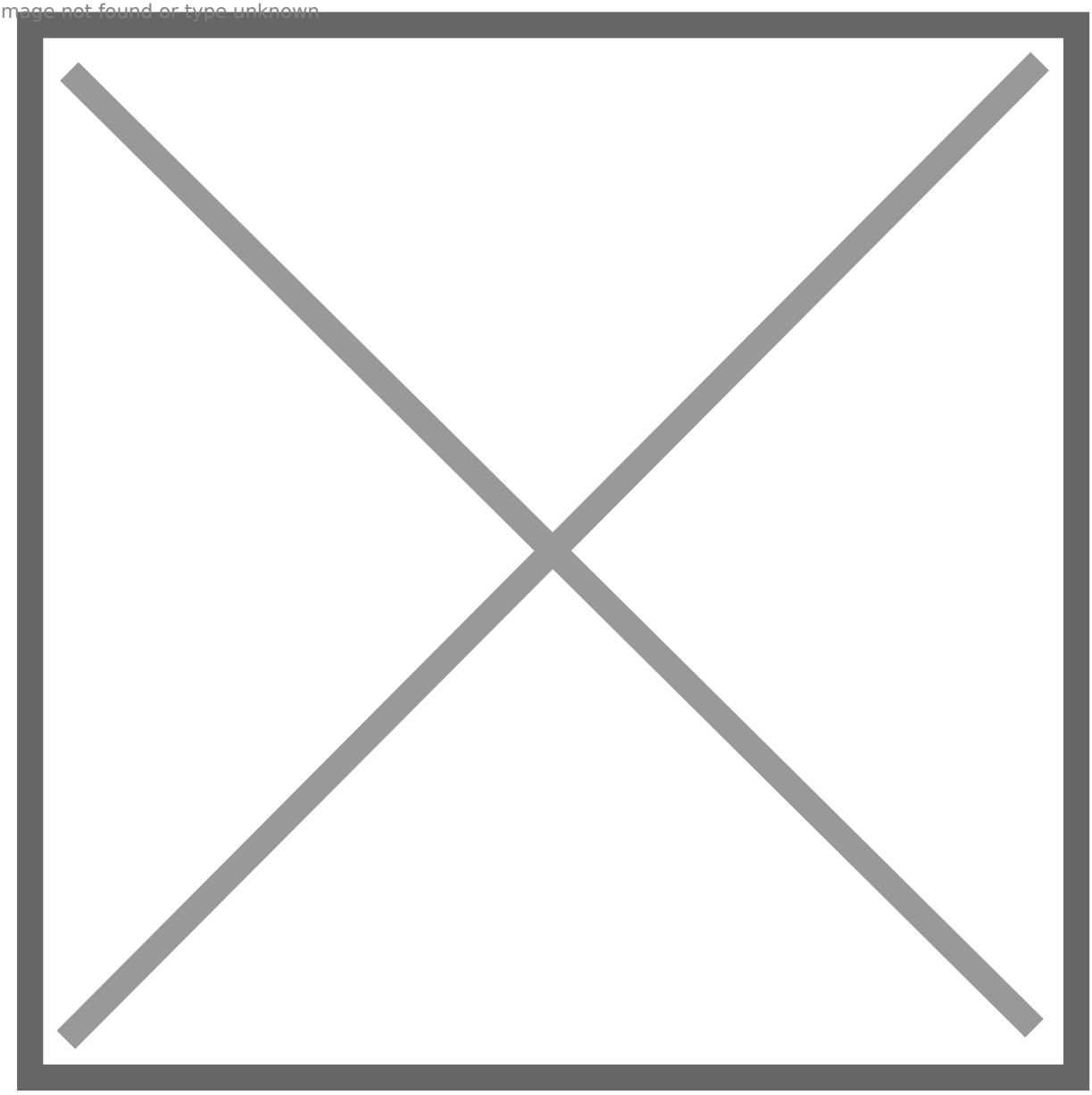
In the query window of SQL Server Management Studio, run the script below to identify the location of the TempDB data and log file:

```
Use master
GO
```

```
SELECT
name AS [LogicalName]
,physical_name AS [Location]
,state_desc AS [Status]
FROM sys.master_files
WHERE database_id = DB_ID(N'tempdb');
GO
```

This query shows us the presence of the TempDB in the default folder of the SQL Server installation :

Image not found or type unknown



TempDB in the default folder

Once you have identified the location of the TempDB files, the next step will be to create folders on the new hard drive where you want to store the TempDB data and log file.

However, you must ensure that the new location where the TempDB files are stored is accessible by SQL Server. That is, you must ensure that the account under which the SQL Server service is

running has read and write permissions to the folder where the files are stored.

Change Location of TempDB Data Files and Log Files Using ALTER DATABASE

Run the ALTER DATABASE command below to change the TempDB data and log file location in SQL Server:

```
USE master;
GO
ALTER DATABASE tempdb
MODIFY FILE (NAME = tempdev, FILENAME = 'T:\MSSQL\DATA\tempdb.mdf');
GO
ALTER DATABASE tempdb
MODIFY FILE (NAME = templog, FILENAME = 'T:\MSSQL\DATA\templog.ldf');
GO
```

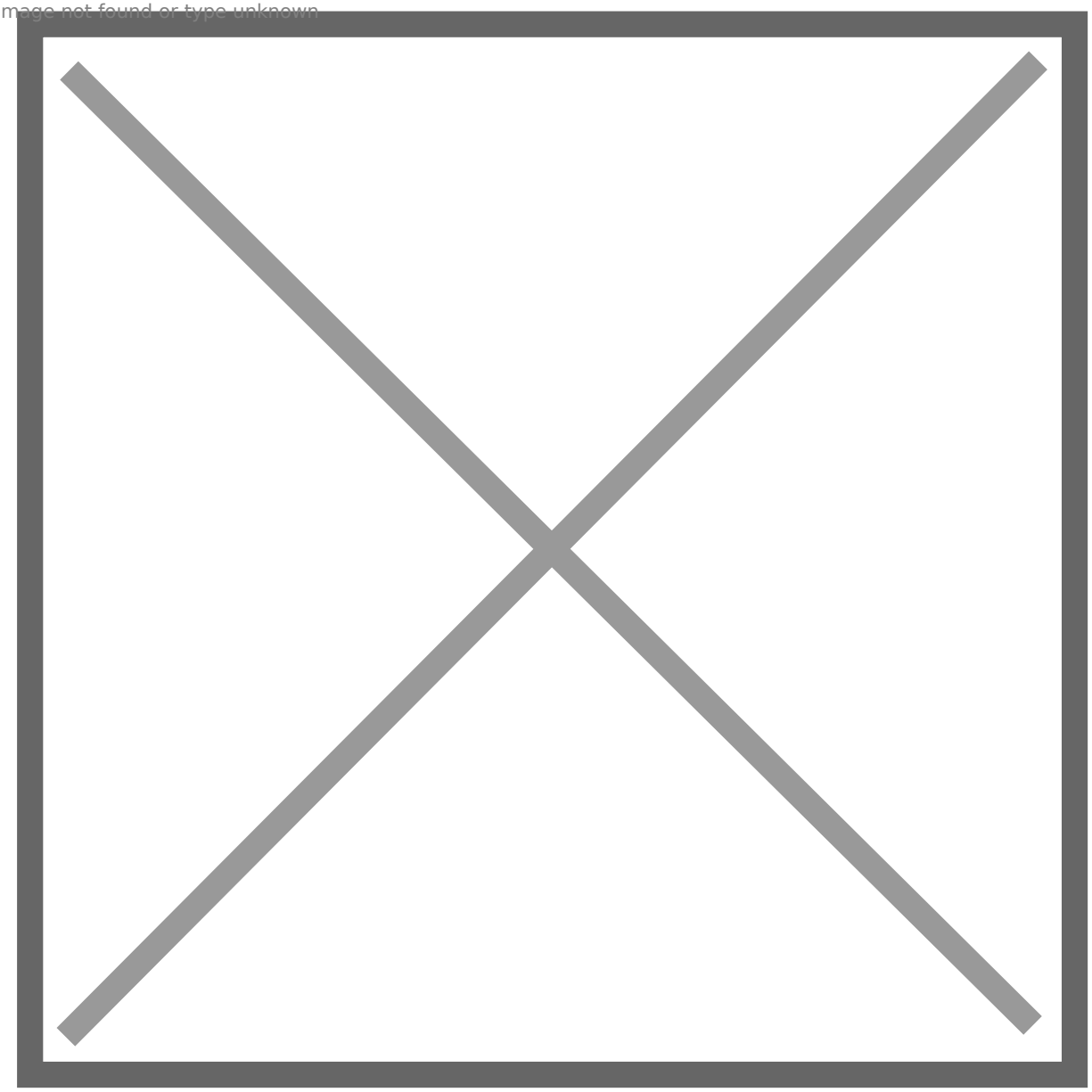
Tip if you have several tempDB files

It happens that sometimes we have several tempDB databases to move, as shown in our example above. Instead of doing several copy / paste, a small script to generate all TempDB move requests:

```
SELECT 'ALTER DATABASE tempdb MODIFY FILE (NAME = [' + f.name + '],'
+ ' FILENAME = ''Z:\MSSQL\DATA\' + f.name
+ CASE WHEN f.type = 1 THEN '.ldf' ELSE '.mdf' END
+ ''');'
FROM sys.master_files f
WHERE f.database_id = DB_ID(N'tempdb');
```

The result is:

Image not found or type unknown



generate all TempDB move requests

A simple copy / paste of all the queries to execute them at once!

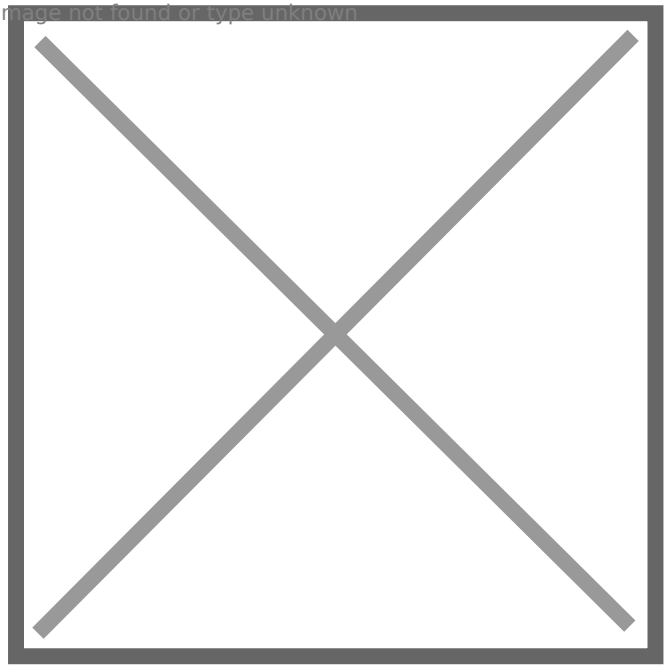
Stop and restart the SQL Server service

Stop and restart the instance of SQL Server for the changes to take effect.

Verify Changed Location of TempDB Data Files and Log Files

All you have to do is re-execute the very first query. Results:

Image not found or type unknown



Delete old tempdb.mdf and templog.ldf files

The final step will be to delete the tempdb.mdf and templog.ldf files from the original location. So just go there to the location and delete them with the DELETE key on your keyboard or with the right mouse button.

> **Note**: SQL Server does not support moving the TempDB database using backup/restore and using database detach methods. The only way to do this is by Transact SQL code as shown above.

MsSQL - Set All User Tables to Simple on DEV/TEST Server

```
/*

This script is to set all of the user database to simple on a DEV/TEST server

Use the @Update and set to 1 for updating or 0 default to ready only

Created By: Steve Ling 2024/09/05

*/

USE MASTER

declare @isql varchar(2000), @dbname varchar(64), @logfile varchar(128), @Update bit

Set @Update = 0 --0=Ready Only, 1=Update Databases

declare c1 cursor for

SELECT d.name, mf.name as logfile--, physical_name AS current_file_location, size
FROM sys.master_files mf
inner join sys.databases d
on mf.database_id = d.database_id
where recovery_model_desc <> 'SIMPLE'
and d.name not in
('master','model','msdb','tempdb','DB_Administration','DWConfiguration','DWDiagnostics','DWQueue')
and mf.type_desc = 'LOG'

open c1
```

```
fetch next from c1 into @dbname, @logfile

While @@fetch_status <> -1

begin

select @isql = 'ALTER DATABASE ' + @dbname + ' SET RECOVERY SIMPLE'

print @isql

If @Update = 1
exec(@isql)

select @isql='USE ' + @dbname + ' checkpoint'

print ''+@isql

If @Update = 1
exec(@isql)

select @isql='USE ' + @dbname + ' DBCC SHRINKFILE (' + @logfile + ', 1)'

print ''+@isql

If @Update = 1
exec(@isql)

fetch next from c1 into @dbname, @logfile

end

close c1

deallocate c1
```