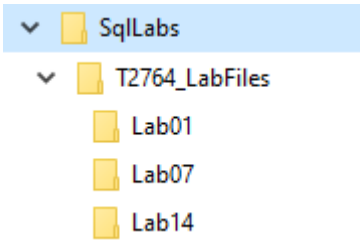
Lab Manual

T2764, Administering a SQL Database

Lab environment overview

- **Tip**: open this lab manual inside the lab virtual machine. This makes it easier to copy and paste from this document.
    - For Virsoft VM environment, pasting from *outside* the VM requires you to right-click the notepad at the top left and select "paste…"
- Your machine name is ***NORTH***.
- If not already done for you, log in to Windows using
    - ***Student***
    - ***myS3cret***
- Copy the lab files to your virtual machine. You find them here: https://karaszi.com/training
- Extract the files, so you in the root of your C: drive end up with a folder structure looking like below



- You probably want to use ***SQL Server Management Studio (SSMS)*** to do the labs. You are welcome to use Azure Data Studio (ADS) as well, but be prepared to do most admin work typing the T-SQL commands (there are less dialogs in ADS to manage SQL Server).
- You will use the following SQL Server instances:
    - ***Default*** (connect using just the machine name ***NORTH***). This is the main instance for your labs.
    - Some labs might use the ***A*** instance (connect using ***NORTH\A***). The lab instructions will specify if that is the case.
- Login to your SQL Server instance using Windows authentication, unless otherwise specified.
    - (There is a Windows login in each instance for your Windows account, which is a sysadmin.)
- To keep things simple, we are *not* in a domain environment, but you can imagine that we are.
    - Use NORTH (the machine name) where you normally have a domain name.

- You can always revert/reset your databases to "default".
    - There are three bat files in the C:\SqlLabs folder that performs a restore of the demo database, one for each database.
    - Note: **Run as Administrator**
- The lab answers are not designed to be used independently. Use the lab instructions and check the answers if needed.

Lab 1: Authentication to SQL Server

Ex 1. Verify Windows groups and accounts

NOTE 1: Before running the lab, run below bat file (**Run as Administrator**)

C:\SqlLabs\T2764_LabFiles\Lab01**Setup.bat**

NOTE 2: We are not in a domain environment, to keep things simple. I.e., specify your machine name instead of domain name: for instance, NORTH\Sue.

Use Computer Management on your machine and verify that below Windows groups and accounts exists:

- DBA
  - Members: John
- ItSupport
  - Members: Frank, Bill, Pat
- Controllers
  - Members: Pat, Jean

Above should have been created by the bat file you ran in the above step. If they don't exist, re-run Setup.bat (as Administrator) or just create the missing groups and accounts (*as **local** groups and accounts*).

Ex 2. Set SQL server to mixed security mode

Use SSMS to verify that your SQL Server instance is in mixed mode security. If it isn't, then change the setting and restart the SQL Server service.

Ex 3. Create logins

Create the following logins:

- Windows logins
  - NORTH\DBA
  - NORTH\ItSupport
  - NORTH\Controllers
- SQL logins (the password should be "myS3cret")
  - SqlTom
  - SqlAnne

Ex 4. Create users for your logins

Create users for below logins in the Adventureworks database:

- SqlTom
- SqlAnne
- NORTH\ItSupport

Create users for below logins in the AdventureworksDW database:

- NORTH\Controllers
- NORTH\ItSupport

Ex 5. If time permits. Manage an orphaned user

Your SQL user Jill in the Adventureworks database ***on the A instance*** is orphaned.

Note: This exercise will be done on NORTH\A.

First, you want to list your orphaned users. Use sp_change_users_login to do this.

And then you want to connect that user to a login. You have several options to do this

1. Create a login for Jill and then use ALTER USER Jill WITH NAME = Jill (or sp_change_users_login).
2. Script out the login from the source instance using sp_help_revlogin (search and install from the internet) and then create the login on the A instance based on the command generated by sp_help_revlogin.
3. Use some other of the many scripts available out there.

Lab 1 answer suggestions

Ex 2. Set SQL server to mixed security mode

SSMS, Object Explorer, right-click your instance, Properties, the Security page, Server Authentication: "SQL Server and Windows authentication mode".

Ex 3. Create logins

```
CREATE LOGIN [NORTH\DBA] FROM WINDOWS
CREATE LOGIN [NORTH\ItSupport] FROM WINDOWS
CREATE LOGIN [NORTH\Controllers] FROM WINDOWS

CREATE LOGIN SqlTom WITH PASSWORD = 'myS3cret'
CREATE LOGIN SqlAnne WITH PASSWORD = 'myS3cret'
```

Ex 4. Create users for your logins

```
USE Adventureworks
CREATE USER SqlTom
CREATE USER SqlAnne
CREATE USER [NORTH\ItSupport]


USE AdventureworksDW
CREATE USER [NORTH\Controllers]
CREATE USER [NORTH\ItSupport]
```

Ex 5. Manage an orphaned user

List the orphaned users, on the NORTH\A instance:

```
USE Adventureworks
EXEC sp_change_users_login @Action = 'Report'
```

Create login for Jill:

```
CREATE LOGIN Jill WITH PASSWORD = 'myS3cret'
```

Map the user for Jill to the correct login:

```
USE Adventureworks
ALTER USER Jill WITH LOGIN = Jill
```

Lab 2: Server and database roles

NOTE 1: Before running the lab, run below bat file (**double-click or right-click, Open**)

C:\SqlLabs\T2764_LabFiles\Lab02**Setup.bat**

NOTE 2: This lab depends on Lab 1, exercise 1, 2 and 4. Run the Answer Suggestions for those if you didn't do them

Ex 1. Server roles and server permissions

Make sure you do this lab on the *default instance* (just as a reminder, you might have done the last lab steps on the A instance).

Assign the sysadmin role to the NORTH\DBA login (which refers to a Windows group).

Create a server role named jrDBA and grant the following server permissions to the role:

- ALTER ANY CREDENTIAL
- ALTER ANY DATABASE
- ALTER SERVER STATE
- VIEW SERVER STATE

Add the NORTH\ItSupport login to the above jrDBA role.

Ex 2. Database roles and database permissions

In the AdventureworksDW database:

- Add the NORTH\Controllers user to the db_datareader role
- Add the NORTH\ItSupport user to the db_owner role

In the Adventureworks database:

- Add the NORTH\ItSupport user to the db_owner role
- Create a database role named DbExec
- Add the following permission to the DbExec role, at the database level
    - EXECUTE
- Add the user SqlTom to the DbExec role
- Add the user SqlAnne to the db_datareader and db_datawriter roles

Ex 3. Test the roles and permissions

In the AdventureworksDW database:

- Use EXECUTE AS LOGIN to run as NORTH\Pat (which is a member of the NORTH\Controllers group in Windows), and verify that she can do SELECT from the DimDate table. Remember to execute the REVERT command when you're done.

In the Adventureworks database:

- Use EXECUTE AS LOGIN to run as SqlTom, and verify that he can execute the GetProductColors stored procedure. (This procedure requires no parameters.)

Lab 2 answer suggestions

Ex 1. Server roles and permissions

```
USE master
GO
ALTER SERVER ROLE sysadmin ADD MEMBER [NORTH\DBA]

CREATE SERVER ROLE jrDBA
GRANT ALTER ANY CREDENTIAL TO jrDBA
GRANT ALTER ANY DATABASE TO jrDBA
GRANT ALTER SERVER STATE TO jrDBA
GRANT VIEW SERVER STATE TO jrDBA

ALTER SERVER ROLE jrDBA ADD MEMBER [NORTH\ItSupport]
```

Ex 2. Database roles and database permissions

```
USE AdventureworksDW
GO
ALTER ROLE db_datareader ADD MEMBER [NORTH\Controllers]
ALTER ROLE db_owner ADD MEMBER [NORTH\ItSupport]

USE Adventureworks
GO
ALTER ROLE db_owner ADD MEMBER [NORTH\ItSupport]
CREATE ROLE DbExec
GRANT EXECUTE TO DbExec
ALTER ROLE DbExec ADD MEMBER SqlTom
ALTER ROLE db_datareader ADD MEMBER SqlAnne
ALTER ROLE db_datawriter ADD MEMBER SqlAnne
```

Ex 3. Test the roles and permissions

```
USE AdventureworksDW
EXECUTE AS LOGIN = 'NORTH\Pat'
SELECT * FROM DimDate
REVERT
GO

USE Adventureworks
EXECUTE AS LOGIN = 'SqlTom'
EXEC GetProductColors
REVERT
GO
```

Lab 3: Granting permissions

NOTE: This lab depends on Lab 1, exercise 1, 2 and 4 and Lab 2 exercise 1 and 2. Run the Answer Suggestions for those if you didn't do them

Ex 1. Grant and deny permissions at object and schema level

In the Adventureworks database:

Grant SELECT permission on the Production.Product *table* to SqlTom

Grant SELECT permission on the Sales *schema* to SqlTom

Deny SELECT permission on the Sales.SalesPersonQuotaHistory *table* from SqlTom

Ex 2. Test above permissions

Use EXECUTE AS LOGIN to execute as SqlTom in the Adventureworks database and test the above assigned permissions, including the DENY.

I.e., SqlTom should be able to SELECT from all tables in the Sales schema, except the Sales.SalesPersonQuotaHistory table. (No need to try all the tables in the schema, but do try a few of them, including the one having DENY.)

Lab 3 answer suggestions

Ex 1. Grant and deny permissions at object and schema level

```
USE Adventureworks

GRANT SELECT ON Production.Product TO SqlTom
GRANT SELECT ON SCHEMA::Sales TO SqlTom
DENY SELECT ON Sales.SalesPersonQuotaHistory TO SqlTom
```

Ex 2. Test above permissions

```
USE Adventureworks
EXECUTE AS LOGIN = 'SqlTom'
SELECT * FROM Production.Product
SELECT * FROM Sales.Customer
GO

--This should fail:
SELECT * FROM Sales.SalesPersonQuotaHistory
GO

REVERT
```

Lab 4: Auditing and encryption

You will most probably not have time to do all three exercises, so do:

- Whichever exercise(s) of below you find interesting
- In what order you want
- As time permits

Ex 1. Implement Server Audit

You will create a server audit to capture login information; and SELECT in the Adventureworks database.

- Create the server audit. Name it myAudit.
- The target should be file, in the folder C:\DemoDatabases
- Feel free to play with the other options, it isn't important for the lab what you set those to.

- Create a server audit specification for above server audit. Name it myAuditServerSpec.
- Add the action group SUCCESSFUL_LOGIN_GROUP
- Make sure that the state is ON

- Create a database audit specification in the Adventureworks database, using the server audit you created earlier. Name it myAuditServerSpec.
- Add the action SELECT and UPDATE on Sales.Customer for the public user (which means everybody).
- Make sure that the state is ON

- Turn on the server audit

- Test below by connect a query windows, as SqlAnne with the password "myS3cret" (make sure you connect to the **default instance**), run the query in the Adventureworks database:
  ```
  SELECT * FROM Sales.Customer AS c WHERE c.StoreID = 930
  ```
- If the connection doesn't work, make sure that SqlAnne has a login, a user in Adventureworks and is member of db_datareader.
- You should be able to right-click the server audit and check if there are any audit records. ***Note***: this will fail with an error and an empty window on 2022 RTM due to a bug in the database engine (returning some duplicate which SSMS errors on). You have two options if you encounter this bug. One is to update your SQL Server to a fresh CU. The other option is to read the trace using a query instead, something like this:

```
SELECT f.event_time, f.succeeded, f.session_id, f.session_server_principal_name,
f.server_principal_name, f.database_principal_name, f.statement
FROM fn_get_audit_file('C:\DmoDatabases\myAudit*', default, default) AS f
WHERE f.session_server_principal_name <> 'NT Service\SQLTELEMETRY'
ORDER BY event_time
```

- Clean up. Disable and then delete the database audit specification, server audit specification and server audit (i.e., in reverse create order).

Ex 2. Implement TDE

You will implement TDE on a newly created database, perform a backup and then restore it on the A instance.

- On the default instance
- Create a database named myDatabase
- Create a service master key. Specify a strong password, for instance 'fjWr.C-pifkqie*qwi8'.
- Backup the service master key to C:\DemoDatabases\smk.bak. Use a strong password, for instance 'sh3fkjnmcteh&ölkdSw.We3'.
- Create a server certificate named myTdeCert, having the subject 'Certificate for TDE'.
- Backup the server certificate to the file C:\DemoDatabases\TDE_cert.bak. Specify that the private key goes to C:\DemoDatabases\TDE_cert_pk.bak having a strong password (for instance 'ssdhj4Aakjlfp6d.kjQ#'). **Take a note of the password, you need this later.**
- Create a database encryption key in the database. Use the AES_256 algorithm and encrypt it using the myTdeCert certificate.
- Activate data encryption. Use the GUI, or the ALTER DATABASE command directly.
- Optionally, check the status by doing SELECT from sys.dm_database_encryption_keys DMV, check the encryption_state column (1 means unencrypted, 2 means in-progress and 3 means encrypted).
- Note that TDE isn't fully activated until you've also emptied the transaction log. You can stress this by execution the CHECKPOINT command in the database if in simple recovery model, or performing a log backup if in full recovery model.
- Detach the myDatabase database.
- Note where the database files are for this database. Use the GUI or SELECT from sys.databases (in the myDatabase database).

- Copy the database files to the C:\DbFiles\A folder

- Connect to the A instance.
- Try to attach the database. You should get an error.
- Create a service master key. Specify a strong password.
- Try restoring the server certificate from the backup file you created earlier. Use the password you specified in the BACKUP CERTIFICATE command above.
- This will likely fail because the A instance don't have access to those backup files. Grant full permissions on the files to the service account for the A instance.
- Re-try restoring the server certificate.
- Re-try attaching the database.
- If attach fails, then the service account for the A instance need full permissions on the database files. Grant that and re-try the attach.

Ex3. Implement Always Encrypted

You will create a “copy” of the HumanResources.Employee table in the Adventureworks database and use Always Encrypted to encrypt the **BirthDate** and **SickLeaveHours** columns. You will use the GUI in SSMS to configure and activate Always Encrypted.

There is no answer suggestion for this exercise.

- Create a “copy” of the HumanResources.Employee in the Adventureworks database table using SELECT INTO. Name the new table myEmployees. See below:
- `SELECT * INTO myEmployees FROM HumanResources.Employee`
- Refresh the Tables folder in SSMS.
- Right-click the table you created above (dbo.myEmployees) and select Encrypt Columns
- Follow the Wizard (the **BirthDate** and **SickLeaveHours** columns). It doesn’t matter if you choose Deterministic or Randomized encryption for the sake of the lab. Choose default for the rest, unless you feel you want to test something specific.
- Use below query to determine that the desired columns are encrypted.
- `SELECT * FROM myEmployees`

Lab 4 answer suggestions

Ex 1. Implement Server Audit

```
USE master

--Create the server audit
CREATE SERVER AUDIT myAudit
TO FILE
(FILEPATH = 'C:\DemoDatabases')
GO

--Create the server audit specification
CREATE SERVER AUDIT SPECIFICATION myAuditServerSpec
FOR SERVER AUDIT myAudit
 ADD (SUCCESSFUL_LOGIN_GROUP)
WITH (STATE = ON);

--Create a database audit specification
USE Adventureworks
GO

CREATE DATABASE AUDIT SPECIFICATION myAuditServerSpec
FOR SERVER AUDIT myAudit
 ADD (SELECT, UPDATE ON Sales.Customer BY public)
WITH (STATE = ON);
GO

--Turn on the server audit
USE master;
GO
ALTER SERVER AUDIT myAudit WITH (STATE = ON);
GO

--Test it. Connect using SqlAnne with the password “myS3cret”
USE Adventureworks
SELECT * FROM Sales.Customer AS c WHERE c.StoreID = 930

--Right-click the server audit and verify the audit records. If the result is empty
due to bug in SQL Server 2022, update your SQL Server to a higher CU or use below
query:
SELECT f.event_time, f.succeeded, f.session_id, f.session_server_principal_name,
f.server_principal_name, f.database_principal_name, f.statement
FROM fn_get_audit_file('C:\DemoDatabases\myAudit*', default, default) AS f
WHERE f.session_server_principal_name <> 'NT Service\SQLTELEMETRY'
ORDER BY event_time


--Clean up. Connect using Windows authentication
USE Adventureworks
ALTER DATABASE AUDIT SPECIFICATION myAuditServerSpec WITH (STATE = OFF);
DROP DATABASE AUDIT SPECIFICATION myAuditServerSpec

USE master
ALTER SERVER AUDIT SPECIFICATION myAuditServerSpec WITH (STATE = OFF);
DROP SERVER AUDIT SPECIFICATION myAuditServerSpec
GO

ALTER SERVER AUDIT myAudit WITH (STATE = OFF);
DROP SERVER AUDIT myAudit
```

```
GO
```

Ex 2. Implement TDE

```
--On the default instance
USE master;
GO

CREATE DATABASE myDatabase

--Create the service master key
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'fjWr.C-pifkqie*qwi8'

--Backup the server master key
BACKUP MASTER KEY TO FILE = 'C:\DemoDatabases\smk.bak'
 ENCRYPTION BY PASSWORD = 'sh3fkjnmcteh&ölkdSw.We3'

--Create a server certificate
CREATE CERTIFICATE myTdeCert WITH SUBJECT = 'Certificate for TDE'

--Backup the server certificate
BACKUP CERTIFICATE myTdeCert
TO FILE = 'C:\DemoDatabases\TDE_cert.bak'
WITH PRIVATE KEY
(FILE = 'C:\DemoDatabases\TDE_cert_pk.bak'
,ENCRYPTION BY PASSWORD = 'ssdhj4Aakjlfp6d.kjQ#')
GO


--Create a database encryption key in the database
USE myDatabase
GO
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_256
ENCRYPTION BY SERVER CERTIFICATE myTdeCert;
GO

--Activate data encryption
ALTER DATABASE myDatabase SET ENCRYPTION ON
GO

--For the sake of the lab, set the database to simple recovery model
ALTER DATABASE myDatabase SET RECOVERY SIMPLE

--Empty the log
CHECKPOINT

--Check encryption status
SELECT d.name, d.is_encrypted, e.encryption_state
FROM sys.dm_database_encryption_keys AS e
 JOIN sys.databases AS d ON d.database_id = e.database_id;
GO

--Check the path for the database files
SELECT physical_name FROM myDatabase.sys.database_files

--Detach the database (make sure that there are 0 connections to the database)
USE master
GO
EXEC sp_detach_db 'myDatabase'
```

```
--Connect to the A instance.

--Copy the database files to the C:\DbFiles\A folder

--Try to attach the database
CREATE DATABASE myDatabase ON
 (FILENAME = N'C:\DbFiles\a\myDatabase.mdf' )
,(FILENAME = N'C:\DbFiles\a\myDatabase_log.ldf' )
 FOR ATTACH
--Above should fail, due to lacking ceritifcate.

--Create the service master key
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'jldSdflkj#.ljkQds!jkl';

--Create the server certificate from backup files
CREATE CERTIFICATE TDE_cert
FROM FILE = 'C:\DemoDatabases\TDE_cert.bak'
WITH PRIVATE KEY
(FILE = 'C:\DemoDatabases\TDE_cert_pk.bak'
,DECRYPTION BY PASSWORD = 'ssdhj4Aakjlfp6d.kjQ#'
);
--Above might fail, due to lacking permissions for the files.


--Run below from OS command prompt (Run as Administrator)
--to grant permissions to the service account
/*
icacls C:\DemoDatabases\TDE_cert.bak /setowner MSSQL$A
icacls C:\DemoDatabases\TDE_cert.bak /grant MSSQL$A:F
icacls C:\DemoDatabases\TDE_cert_pk.bak /setowner MSSQL$A
icacls C:\DemoDatabases\TDE_cert_pk.bak /grant MSSQL$A:F
*/

--Re-try the CREATE CERTIFICATE command

--Re-try the attach of the database.
--Above might fail, due to lacking permissions for the files.

--Run below from OS command prompt to grant permissions to the service account
/*
icacls C:\DbFiles\a\myDatabase.mdf /setowner MSSQL$A
icacls C:\DbFiles\a\myDatabase.mdf /grant MSSQL$A:F
icacls C:\DbFiles\a\myDatabase_log.ldf /setowner MSSQL$A
icacls C:\DbFiles\a\myDatabase_log.ldf /grant MSSQL$A:F
*/

--Re-try the CREATE CERTIFICATE and attach of the database.
```

Ex3. Implement Always Encrypted

There is no answer suggestion for this exercise. Use the lab instructions.

Lab 6: Backup

Ex 1. Setup default backup options

- Create the folder C:\SqlBackups
- Configure the default backup folder to be C:\SqlBackups
- Configure so that backups are compressed by default
- Configure so that a backup checksum is included in the backup header by default

Ex 2. Perform different backup types

Do the backups to the C:\SqlBackups folder.

- Verify the default backup folder by executing below (note that there is *no* backup path specified, only the file name). You can also open the backup dialog see what path the file destination dialog points to. You might have to restart the database engine and/or SSMS before the reconfigured path takes place.
```
BACKUP DATABASE Adventureworks TO DISK = 'Adventureworks.bak'
```
- Perform a full backup of Adventureworks to Adventureworks.bak filename. Use default backup options (whether you use the GUI or the BACKUP command directly).
- Perform one more full backup of Adventureworks to Adventureworks.bak filename. Again, use default backup options.
- Use RESTORE HEADERONLY to verify that above backup file now includes two or three full backups (depending if you did all three above to the same file name).
- Perform a full backup of Adventureworks to Adventureworks.bak filename. Overwrite the contents on that file.
- Use RESTORE HEADERONLY to verify that now there's only one backup in the file.
- Perform a differential backup to the AdventureworksDiff.bak filename.
- Perform a log backup to the Adventureworks.trn filename.
- You probably got an error from above command. Why?
- Set the database to FULL recovery model.
- Try to do a log backup now. Did it succeed? Why not?
- Perform yet another full backup (to any filename) and re-try the log backup. Now it should have executed successfully.

Lab 6 answer suggestions

Ex 1. Setup default backup options

- Create the folder C:\SqlBackups
- Configure the default backup folder to be C:\SqlBackups
  - Object Explorer, right-click the instance, Properties, Database Settings
- Configure so that backups are compressed by default
  - Same dialog as above or below T-SQL

```
EXEC sp_configure 'backup compression default', '1'
RECONFIGURE
```

- Configure so that a backup checksum is included in the backup header by default
  - Same dialog as above or below T-SQL

```
EXEC sp_configure 'backup checksum default', '1'
RECONFIGURE
```

Ex 2. Perform different backup types

```
BACKUP DATABASE Adventureworks TO DISK = 'Adventureworks.bak'

BACKUP DATABASE Adventureworks TO DISK = 'C:\SqlBackups\Adventureworks.bak'

BACKUP DATABASE Adventureworks TO DISK = 'C:\SqlBackups\Adventureworks.bak'

RESTORE HEADERONLY FROM DISK = 'C:\SqlBackups\Adventureworks.bak'

BACKUP DATABASE Adventureworks TO DISK = 'C:\SqlBackups\Adventureworks.bak'
WITH INIT

RESTORE HEADERONLY FROM DISK = 'C:\SqlBackups\Adventureworks.bak'

BACKUP DATABASE Adventureworks TO DISK = 'C:\SqlBackups\AdventureworksDiff.bak'
WITH DIFFERENTIAL

BACKUP LOG Adventureworks TO DISK = 'C:\SqlBackups\Adventureworks.trn'

ALTER DATABASE Adventureworks SET RECOVERY FULL

BACKUP LOG Adventureworks TO DISK = 'C:\SqlBackups\Adventureworks.trn'

BACKUP DATABASE Adventureworks TO DISK = 'C:\SqlBackups\Adventureworks.bak'

BACKUP LOG Adventureworks TO DISK = 'C:\SqlBackups\Adventureworks.trn'
```

Lab 7: Restore

NOTE: Before running the lab, run below bat file (**Run as Administrator**)

C:\SqlLabs\T2764_LabFiles\Lab07**Setup.bat**

Tip: Above restores the red, white and blue databases. It also breaks the blue database, so some error messages are expected. In case that the restore fails or "hangs", then there are probably connections to that database. That shouldn't happen since the bat file stops and starts the default instance (assuming you do Run as Administrator). If that still happens, for whatever reason, you can investigate which connections are connected to any of these databases by executing "sp_who", and kick out those connections using "KILL <spid>". Or use "ALTER DATABASE dbname SET SINGLE_USER WITH ROLLBACK IMMEDIATE".

Verify that you have the three red, white and blue databases. The blue database should have a weird status, probably "Recovery Pending".

Ex 1. Restore of an existing, broken, database

The ***blue*** database is broken. You cannot access it. You will restore a backup for this database.

- Try to access the blue database. It should be inaccessible.
- Check the errorlog for error messages, if you want.
- Restore the blue database from the backup file: C:\SqlLabs\T2764_LabFiles\Lab07\blue.bak
- Verify that you can access the database.
- Verify that there are 15 rows in the Persons table.

Ex 2. Restore of a non-existing database

You have been given a backup of the ***red*** database. You will investigate that backup file and then restore it.

- Investigate which backups exists on the C:\SqlLabs\T2764_LabFiles\Lab07\red.bak backup file, using the RESTORE HEADERONLY command.
- Optional: investigate if the backup was striped, using RESTORE LABELONLY.
- Investigate what files the database was using when that backup was produced, using RESTORE FILELISTONLY
- Perform a RESTORE for the database. Make sure that the database files are created in the C:\DbFiles\MsSqlServer folder.
- Verify that there is data (20 rows) in the Persons table in the red database.

Ex 3. Perform a backup. Do something bad. Do restore.

You will work with the ***white*** database. There are currently 15 rows in the Persons table, all with consecutive numbering from 1 to 15 for the PersonID column.

- Perform a backup of the database.
- Do an "accidental" delete of persons 1 to 10 (so you only have 5 rows left in the table).:
  ```
  DELETE FROM Persons WHERE PersonID BETWEEN 1 AND 10
  ```
- Verify that you only have 5 rows in the table
- Perform a restore of above backup. SQL Server might want to you produce a tail-log backup. We don't want that for this exercise, so use the REPLACE option for the RESTORE command.
- Verify that you have 15 rows in the table

Ex 4. If time permits: add log backups to above exercise

Now you will throw in log backups in above scenario and restore to some desired point in time. You are still working with the ***white*** database. For this lab, you are more on your own so you might want to do this where you have plenty of time on your hands and you can experiment.

- Set the database in full recovery model
- Perform a backup of the database.
- Delete persons 1 to 10 (so you only have 5 rows left in the table).:
  ```
  DELETE FROM Persons WHERE PersonID BETWEEN 1 AND 10
  ```
- Verify that you only have 5 rows in the table
- Perform a log backup.
- Note datetime, delete PersonID 11, verify you have 4 rows in the table.
- Perform a log backup.
- Note datetime, delete PersonID 12, verify you have 3 rows in the table.
- Perform a log backup.
- Now do a restore to whatever state you want. For the first RESTORE DATABASE can add the REPLACE option if you don't want to produce a log backup. Also, you can use the STANDBY option if you want to SELECT for the table and still restore more (log) backups.

Lab 7 answer suggestions

Ex 1. Restore of an existing, broken, database

```
--Try to access the database, should fail
USE blue
GO

--Restore the blue database
RESTORE DATABASE blue FROM DISK = 'C:\SqlLabs\T2764_LabFiles\Lab07\blue.bak'
GO

--Verify data
USE blue
SELECT * FROM Persons
```

Ex 2. Restore of a non-existing database

```
--Investigate which backups exists
RESTORE HEADERONLY FROM DISK = 'C:\SqlLabs\T2764_LabFiles\Lab07\red.bak'
--There should be one backup on this file

--Investigate if the backup was striped using RESTORE LABELONLY.
RESTORE LABELONLY FROM DISK = 'C:\SqlLabs\T2764_LabFiles\Lab07\red.bak'
--The "FamilyCount" column should be 1, meaning the backup wasn't striped

--Investigate what files the database was using when that backup was produced
RESTORE FILELISTONLY FROM DISK = 'C:\SqlLabs\T2764_LabFiles\Lab07\red.bak'
WITH FILE = 1

--Perform the RESTORE
RESTORE DATABASE red FROM DISK = 'C:\SqlLabs\T2764_LabFiles\Lab07\red.bak'
WITH
FILE = 1
,MOVE 'red' TO 'C:\DbFiles\MsSqlServer\red.mdf'
,MOVE 'red_log' TO 'C:\DbFiles\MsSqlServer\red_log.ldf'

--Verify that we have data
SELECT * FROM red..Persons
```

Ex 3. Produce a backup. Do something bad. Perform restore.

```
USE master
BACKUP DATABASE white TO DISK = 'C:\SqlBackups\white.bak' WITH INIT

--Should be 15 rows
SELECT * FROM white..Persons

--Delete PersonID 1 to 10
DELETE FROM white..Persons WHERE PersonID BETWEEN 1 AND 10

--Should be 5 rows
SELECT * FROM white..Persons

--Restore the database
RESTORE DATABASE white FROM DISK = 'C:\SqlBackups\white.bak' WITH REPLACE

--Should be 15 rows
SELECT * FROM white..Persons
```

Ex 4. If time permits: add log backups to above exercise

```
USE master
ALTER DATABASE white SET RECOVERY FULL
BACKUP DATABASE white TO DISK = 'C:\SqlBackups\white.bak' WITH INIT

--Check out the data, should be 15 rows
SELECT * FROM white..Persons

--Delete PersonID 1 - 10
DELETE FROM white..Persons WHERE PersonID BETWEEN 1 AND 10

--Check out the data, should be 5 rows
SELECT * FROM white..Persons

--Perform a log backup
BACKUP LOG white TO DISK = 'C:\SqlBackups\white_1.trn' WITH INIT

--Note datetime and delete PersonID 11
SELECT GETDATE() --2022-06-30 12:03:26.957
DELETE FROM white..Persons WHERE PersonID = 11
SELECT * FROM white..Persons --4 rows in table

--Perform another log backup
BACKUP LOG white TO DISK = 'C:\SqlBackups\white_2.trn' WITH INIT

--Note datetime and delete PersonID 12
SELECT GETDATE() --2022-06-30 12:03:37.407
DELETE FROM white..Persons WHERE PersonID = 12
SELECT * FROM white..Persons

--Note datetime and delete PersonID 13
SELECT GETDATE() --2022-06-30 12:03:47.627
DELETE FROM white..Persons WHERE PersonID = 13
SELECT * FROM white..Persons

--Perform another log backup
BACKUP LOG white TO DISK = 'C:\SqlBackups\white_3.trn' WITH INIT

--Restore the database.
--We don't care about the tail-log backup so we use the REPLACE option.
RESTORE DATABASE white FROM DISK = 'C:\SqlBackups\white.bak'
WITH NORECOVERY, FILE = 1, REPLACE

--Now use the log backups and restore the database to whatever state you desire
--Example:
RESTORE LOG white FROM DISK = 'C:\SqlBackups\white_1.trn'
WITH NORECOVERY, FILE = 1

RESTORE LOG white FROM DISK = 'C:\SqlBackups\white_2.trn'
WITH NORECOVERY, FILE = 1

RESTORE LOG white FROM DISK = 'C:\SqlBackups\white_3.trn'
WITH RECOVERY, FILE = 1, STOPAT = '2022-06-30 12:03:37.407'

--See what data you have
SELECT * FROM white..Persons
```

Lab 8: SQL Server Agent jobs

Ex 1: Create a job

You will create a job to produce a backup of the AdventureworksLT database. There will be a deliberate misspelling in the path for the backup file. Use whatever name you want for the job, job step, schedule etc.

- Make sure that the Agent service is running for both default and A instance.
- Create a new job.
- Add a job T-SQL step having the below SQL command. Note the misspelling of the file path.

```
BACKUP DATABASE AdventureworksLT
TO DISK = 'C:\DemoBatabases\AdventureworksLT.bak' WITH INIT
```

- Add an output file for above job step

Ex 2: Run and troubleshoot the job

- Right-click the job to manually execute it
- Note the error message
- Right-click the job and find the error message for the job step
- Also check the output file for error message
- Fix the job and re-execute it

Ex 3: Schedule the job

- Schedule the job to be executed every 5 minutes.
- Go back to this job after a while and verify that the job has executed as you have scheduled.
- If time permits, replace the output file name with an agent token to get a file name based on the computer name. Hint, try the token with PRINT in a job step first, so you know you get the right token.
- Note, this job will now execute every 5 minutes, until you delete or disable the job. The backup will overwrite assuming you specified INIT (as suggested in Ex1). I.e., the backups will not fill the disk. But feel free to disable or delete the job when you feel that you are done with it.

Ex 4: If time permits: create an MSX environment

You will create an MSX environment where the default instance is the Master and the A instance is the Target. You can leave the service accounts as they are. Normally Virtual Service Account won't do, but in this case they will since the instances are on the same machine. A tip is to remember that both the database engine and also Agent has their respective output files where you can see error messages (errorlog and sqlagent.out).

- Create a login on the default instance for the service account that the target Agent is using
    - Create user for that login in the msdb database
    - Make that user a member of the TargerServersRole role
- Enable that the target Agent doesn't require certificate-based encryption of the connection (modify the registry for target Agent). Re-start Agent on the target server after making that change.
- Make sure that both database engine and also Agent services are started.
- Build the MSX environment, having the default instance as master and A as target. You can ignore the email address.
- On the master server, create a very simple job, for instance with a T-SQL job-step executing:
  ```
  PRINT 'Hello world of MSX'
  ```
- Schedule the job for execution every minute
- Specify the A instance as a target server
- Wait a couple of minutes and see if the job was created on the target.
- Check job history on both the target and the master server.

Lab 8 answer suggestions

There are no answer suggestions for this lab. Use the lab instructions. Try to find where in SSMS the dialogs to create jobs etc are. Be careful to check error messages if things fail. And, of course, ask the instructor when needed!

Lab 9: SQL Server Agent security

You will create an Agent job that will fail for security reasons. You will identify why it fails and determine what resolutions there might be. You will then implement one of the resolutions.

Ex 1: Create an Agent job with a security issue

- On the ***default*** instance, create an Agent job with one CmdExec job-step containing: SQLCMD /S NORTH\A /Q "PRINT 'This is my super-advanced job'"
- Specify an output file for the jobstep.
- Execute the job. It should fail. Why did it fail? What are your options to fix it?

Ex 2: Fix the problem using a Credential and an Agent Proxy

- Create a windows account to be used for this jobstep.
- Create a login on the A instance for above account.
- Do below steps on the default instance:
    - Create a Credential for this account.
    - Create an Agent proxy. Allow this Agent proxy to execute CMDExec jobsteps. Add so that the login you are using to work on the default instance has permissions to use the Proxy.
        - Technically this last step isn't necessary since you are a sysadmin, but it would have been necessary for a non-sysadmin.
    - Re-configure the jobstep for your job to use this proxy.
    - Run the job. The job should run successfully. If it fails, then troubleshoot it.

Lab 9 answer suggestions

Ex 1: Create an Agent job with a security issue

The reason it should fail is that:

- the job is executed by the **default instance Agent's service account**.
- It tries to logon to the "**NORTH\A" instance database engine** using Windows authentication.
- There is no login on the A instance for the default instances agent's service account.

Ex 2: Fix the problem using a Credential and an Agent Proxy

There are no answer suggestions for this exercise. Use the lab instructions. Try to find where in SSMS the relevant dialogs are. Be careful to check error messages if things fail. And, of course, ask the instructor when needed!

Lab 10: SQL Server errors and Agent Operators

Ex 1: Create and test an Agent Operator

We will not add the complexity of having an email configuration in this environment. You will create an Operator, even though we didn't configure Database Mail, and then troubleshoot why you didn't get an email from a scheduled job.

- Create an operator. Specify whatever name you want, and a dummy email address (like for instance joe@a.b).
- Create an Agent Job, named for instance TestMail. Add a T-SQL jobstep with for instance below command
  ```
  PRINT 'Hello Agent operator'
  ```
- Notify your operator "When the job completes".
- Right-click and execute the job.
- You obviously didn't get an email since we didn't configure Database Mail.
- Can you see in the job history that the email wasn't sent?
- Check the sqlagent.out file (the Agent log file) for error messages.

Ex 2: Investigate an SQL Server error message

You will execute an SQL command, notice the error message and find it in sys.messages, the errorlog file and also the eventlog.

- Execute below command
  ```
  BACKUP DATABASE myDatabaseX TO DISK = 'nul'
  ```
- What error message(s) do you get in the Query Messages window?
- Look up above error(s) in sys.messages. Modify the IN clause below with as many error numbers as you get from the above failed backup command:

  ```
  SELECT * FROM sys.messages
  WHERE language_id = 1033
   AND message_id IN(xxx, yyy)
  ```
- Note that these are not written to the logs (by default).
- Check the errorlog and see if you got any messages from the failed backup command.
- Do the same with the Windows event log.
- What error message number(s) did you find in the logs from the failed backup command?
- You have encounter one of many inconsistencies in SQL Server. Unfortunately, it isn't always as clean as one would hope. In this case two errors were returned from the command and some *other* error was written to the logs.

Ex3: Create an Agent Event Alert

You will create an Event Alert for above error and see if it fires when you execute the command.

- Create an Event Alert for error number 3041.
- Notify the Operator you created earlier. We know that the notification will fail, but you will see that Agent tried to do a notification in the SQLAGENT.OUT file.
- Execute the backup command given in exercise 2.
- Check the history for the Alert. Note that it should have had at least one occurrence.
- Check the SQLAGENT.OUT file for the failed email notification.

Lab 10 answer suggestions

There are no answer suggestions for this lab. Use the lab instructions. Try to find where in SSMS the relevant dialogs are. Be careful to check error messages if things fail. And, of course, ask the instructor when needed!

Lab 12: Extended Events

Ex 1: Capturing SQL commands from an application

You want to know what SQL is submitted by a client application when a certain action is taken in the application. For the purpose of this lab, we will use Activity Monitor in SSMS as the client application. You can use the "New Session Wizard" or "New Session…" to configure the trace, as you feel comfortable with.

- Create a new trace.
- Do not use a template.
- Add the sql_statement_starting event
- No extra columns (actions) or filters (predicates)
- Configure a file target, for instance to the C:\DemoDatabases folder
- Start the trace
- Hook up the "Watch Live Data" to the trace window
- Open Activity Monitor in SSMS (right-click the instance in Object Explorer)
- Return to the Live Window
- Stop the data Feed
- Scroll through the SQL commands
- Expand your trace and open the event file (view target data)
- Stop the trace

Ex 2: Modify the trace definition

You realize that above wasn't really what you wanted. You want to see the cost of the events, and also see the database name, so you can group by it. You want to keep trace you defined above, so you want to "copy" it and modify that "copy".

- Script the trace definition (as Create) you created in Ex 1
- Modify the name of the trace and also the file name
- Execute the CREATE command
- Modify the trace you just created
- Remove the sql_statement_starting event
- Add the sql_statement_completed event
- Add the action database_name to this event
- Save the trace, start it, let Activity Monitor run for a few minutes with the trace started
- Stop the trace.
- For the file target, do View Target Data
- Select any event
- Add the database_name field as a column to above table
- Also add the duration and row_count fields as columns
- Group by the database_name
- Sum over duration
- You can now see the sum of duration for each database represented in the trace
- Stop the trace, if it isn't stopped already

Ex 3: If time permits: capturing strange events

This is a more informal exercise, do it if you feel like it and use your imagination.

- Create the “Looking for Strange” trace that you find below: https://karaszi.com/looking-for-strange
- Start the trace
- Let it run for a while
    - Have Activity Monitor open
    - Play around in Objects explorer
    - For example:
        - Modify an Agent job
        - Open Design for a table in some database
        - Create a database diagrams
- When you are happy with generated activity on your SQL Server:
- View Target Data for the event_counter target
- Did you get any events? Anything that means something to you?

Lab 12 answer suggestions

Ex 1: Capturing SQL commands from an application

```
--Create a new trace
CREATE EVENT SESSION [Capture SQL starting events] ON SERVER
ADD EVENT sqlserver.sql_statement_starting
ADD TARGET package0.event_file
(SET filename=N'C:\DemoDatabases\Capture SQL starting events')
GO

--Start the trace
ALTER EVENT SESSION [Capture SQL starting events] ON SERVER
STATE = START
```

- Hook up the “Watch Live Data” to the trace window
- Open Activity Monitor in SSMS (right-click the instance in Object Explorer)
- Return to the Live Window
- Stop the data Feed
- Scroll through the SQL commands
- Expand your trace and open the event file

```
--Stop the trace
ALTER EVENT SESSION [Capture SQL starting events] ON SERVER
STATE = STOP
```

Ex 2: Modify the trace definition

```
--Create the new (modified) trace definition
CREATE EVENT SESSION [Capture SQL completed events] ON SERVER
ADD EVENT sqlserver.sp_statement_completed(
    ACTION(sqlserver.database_name))
ADD TARGET package0.event_file
(SET filename=N'C:\DemoDatabases\Capture SQL completed events')
GO

--Start the trace
ALTER EVENT SESSION [Capture SQL completed events] ON SERVER
STATE = START
```

- Let Activity Monitor run for a while with the trace started
- Stop the trace.
- For the file target, do View Target Data
- Select any event
- Add the database_name field as a column to above table
- Also add the duration and row_count fields as columns
- Group by the database_name
- Sum over duration
- You can now see the sum of duration for each database represented in the trace

```
--Stop the trace
ALTER EVENT SESSION [Capture SQL completed events] ON SERVER
STATE = STOP
```

Ex 3: If time permits: capturing strange events

There are no answer suggestions for this exercise. Use the lab instructions.

Lab 13: Monitoring SQL Server

You will probably not have time to do all three exercises, so do:

- Whichever exercise(s) of below you find interesting
- In what order you want
- As time permits

Ex 1: Monitor and resolve a blocking situation

You will create a blocking situation, find more information about it and then decide which connection to KILL to resolve the situation. Note that KILL shouldn't be taken lightly, so use with care in real life!

- A tip is to close all query windows first.
- Open 3 query windows in the Adventureworks database. Let's call then QW1, QW2 and QW3.
- You will work with the Person.Person table
- In QW1
    - Start a transaction
    - Update BusinessEntityID = 1, and set FirstName = Kenneth
    - Leave this query window open
- In QW2 and QW3 (you will do the same thing in both those query windows)
    - Select all rows and columns
- Now QW2 and QW3 should be blocked
- Use Activity Monitor to check out the blocking situation
- Which SessionIDs are blocked?
- Which SessionID is the root cause to the blocking situation?
- Use sp_who and sp_who2 to get that same information
- You can also select from sys.dm_tran_locks. (Note that you will get lots of rows, the vast majority comes from the XML columns and index on that column.)
- Terminate the session which is the root cause to the blocking situation
- Verify that QW2 and QW3 now have received the data
- Optionally download sp_whoisactive and give it a try
- Close all query windows

Ex 2: Use Glenn Berry's diagnostic queries to diagnose your instance

Download the diagnostic scripts. Use a search engine to find them (at the time of writing this lab they can be found at: https://glennsqlperformance.com/resources/).

- What version and CU level of SQL Server you have
- Do you have any server configuration options that differs from Glenn's recommendation? (Feel free to change the config option to Glenn's recommendation if you want. Read about it first in the documentation, unless you know what it means.)
- How much memory is your SQL Server currently using?
- Do you have any database with "too many" VLFs?
- Which database uses the most:
    - CPU
    - I/O
    - Memory
- The top 5 wait types. Note that without much load, you might not get many rows from that query. Consider removing the HAVING and re-run the query.

Ex 3: Use sp_blitz to check for best practices on your instance

Download, install and run sp_blitz. Do you have any findings that you would take action on, if this was a production instance of yours?

Lab 13 answer suggestions

Ex 1: Monitor and resolve a blocking situation

- Open 3 query windows in the Adventureworks database. Let's call then QW1, QW2 and QW3.
- You will work with the Person.Person table
- In QW1
    - Start a transaction
    - Update BusinessEntityID = 1 and set FirstName = Kenneth
    - Leave this query window open

```
USE Adventureworks
BEGIN TRAN
UPDATE Person.Person
SET FirstName = 'Kenneth'
WHERE BusinessEntityID = 1
```

- In QW2 and QW3 (you will do the same thing in both those query windows)
    - Select all rows and columns

```
USE Adventureworks
SELECT * FROM Person.Person
```

- Now QW2 and QW3 should be blocked
- Use Activity Monitor to check out the blocking situation
- Which SessionIDs are blocked?
- Which SessionID is the root cause to the blocking situation?
- Use sp_who and sp_who2 to get that same information

```
EXEC sp_who
EXEC sp_who2
```

- You can also select from sys.dm_tran_locks. (Note that you will get lots of rows, the vast majority comes from the XML columns and index on that column.)

```
SELECT *
FROM sys.dm_tran_locks
```

- Terminate the session which is the root cause to the blocking situation
- `KILL <spid_number>`
- Verify that QW2 and QW3 now have received the data
- Close all query windows

Ex 2: Use Glenn Berry's diagnostic queries to diagnose your instance

You are below given the Query number in Glenn's script, in the order according to the lab instructions (note that Glenn might re-number the queries at any time):

- Query 1
- Query 4
- Query 6
- Query 37
- Query 38
- Query 39
- Query 40
- Query 42

Ex 3: Use sp_blitz to check for best practices on your instance

There are no answer suggestions for this exercise. Use the lab instructions.

Lab 14: Troubleshooting

NOTE: Before running the lab, run below bat file (**Run as Administrator**)

C:\SqlLabs\T2764_LabFiles\Lab14**Setup.bat**

Do not peek at what the files does. You don't want to spoil the fun, do you?

Ex 1: Handle a failed connection situation

You have an application, which seems to fail the connection to SQL Server. It should connect to your default instance. In our case, it is below bat file:

C:\SqlLabs\T2764_LabFiles\Lab14\MakeInvoices.bat

- Troubleshot and fix the issue

Ex 2: Handle an unstable SQL Server instance situation

Try to start the A instance. It might start but act weird, or it might not even start This is a trickier lab.

- Troubleshot and fix the issue

Lab 14 answer suggestions

Ex 1: Handle a failed connection situation

- The issue is twofold:
  - The bat files try to connect using an incorrect password.
    - This can be seen in the errorlog file.
    - The password should be “myS3cret”.
    - Change the password in the bat file to above password
  - The login is disable
    - Enable the login

Ex 2: Handle an unstable SQL Server instance situation

- The issue is that a configuration option is way off, which is why SQL Server won’t start (or is unstable).
- If the SQL Server process is running, you might have to kill it in in the OS (using for instance Task Manager).
- If you try to start the service, it might fail after a while.
- Check the errorlog file, you see memory related messages.
- Start SQL server using the -f switch (using for instance Services).
- Make sure nothing else is connected. Terminate all potentially relevant services, SSMS, Agent etc.
- Connect using SQLCMD.EXE to the A instance and execute below

```
EXEC sp_configure 'max server memory', 14000
GO
RECONFIGURE
GO
SHUTDOWN
GO
```

- Start the A instance

Lab 15: Exporting data

Ex 1: Use BCP to export to comma-separated file

Export all rows from the Production.Product table in Adventureworks to a file named Products.csv in the C:\DemoDatabases folder.

- The table should be a readable text file (Unicode or non-Unicode)
- Columns should be separated by comma
- Row-separator should be separated by CRLF
- Perform the export and open the file in notepad
- (You can disregard the warning about empty strings)

Ex 2: Create a BACPAC file

Create a BACPAC file from the AdventureworksLT database.

- Name the file AdventureworksLT.BACPAC, in the C:\DemoDatabases folder
- Include everything

Lab 15 answer suggestions

Ex 1: Use BCP to export to comma-separated file

BCP Adventureworks.Production.Product out C:\DemoDatabases\Products.csv /c /t, /r\n /T

(You can disregard the warning about empty strings.)

Ex 2: Create a BACPAC file

- Object Explorer, Right-click AdventureworksLT, Tasks, “Export Data-tier Application…”