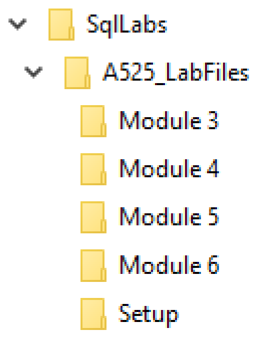
Lab Manual

A525, Upgrade and sharpen your DBA and developer skills

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



```
SqlLabs
  A525_LabFiles
    Module 3
    Module 4
    Module 5
    Module 6
    Setup
```

- 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 your SQL Server 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 2: Installing SQL Server

Ex 1. Before the installation

You will use the SQL Server installation program to view hardware and software requirements, use the configuration checker and also see what is already installed on the machine:

Run setup.exe. It is likely as the D: drive (an attached ISO) (or perhaps as a different drive; or possibly in a folder named something like C:\SqlInstall)

Use the Planning page and view Hardware and Software Requirements.

Use the Tools page and use System Configuration Checker.

Also on the Tools page, check what is installed using the "Installed SQL Server features discovery report".

Ex 2. Install a named instance

You will install a new instance of the database engine. We suggest that you install only the database engine, to keep it quick and simple. (If there is something specific you want to play with on your own during these days you can of course add that to the installation.) You can pretty much do whatever you feel like regarding the settings for the installation since we won't be using this instance in the upcoming labs. Feel free to try things out if you want to. If you prefer some guidelines, you can use below:

- Instance name X
- Database engine only, none of the subcomponents.
- Use default for all folders.
- Use default for all service account (i.e., use Virtual Service Accounts).
- Set so that the database engine and Agent service starts manually.
- Use whatever collation you want. Feel free to select some other collation then what setup defaults to, if you want.
- If you select "Mixed Mode" for security:
    - Password for sa: myS3cret
- Add yourself as an admin login to your SQL Server.

Ex 3. Check the installation

Verify that your instance is started, start it if necessary.

Login to your instance using SSMS or ADS.

Use SQL Server Configuration Manager and configure your instance so it also listens to the TCP network library.

Feel free to stop the instance, if you want to save on some machine resources. But feel free to keep it running if you aim to play with it.

Lab 2 answer suggestions

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

Lab 3: Security

Do whichever exercise(s) you feel for and in whatever order you want.

Ex 1. Containment

1. Configure your SQL Server to accept both SQL Server logins as well as Windows logins:
    a. In Object Explorer, right-click your instance (top), Properties, Security, to the right select "SQL Server And Windows Authentication mode"
2. Re-start you SQL Server (Services, or right-click it in Object Explorer)
3. Allow contained authentication as the instance level (sp_configure)
4. Create a new database, details don't matter.
5. Make it contained.
6. Create an SQL login with password.
7. Connect with Object Explorer to the contained database.
8. From there, open a query window.
9. Try a USE command to some other database.

Ex 2. Data classification and vulnerability report

1. Do a data classification in the Adventureworks databases, from SSMS.
2. Study the recommendations for a short while, just to get an idea.
3. Accept all recommendations and save them.
4. Generate a report, based on the data classification you just did.
5. Again, study it for a short while to get an idea what it is all about.
6. Do a Vulnerability Assessement.
7. Check out the failures. Which would you act on?
8. Check out the passed rules. Not in details, just a quick look.

Ex 3. Dynamic Data Masking

1. Run the "Create Persons Table.sql".
2. You now have a table called Persons
3. Create a user without login, so we have something to play with.
4. Grant above user SELECT permission on the Persons table.
5. Mask the Persons table according to below
   a. Don't expose the real BirthDate. Doesn't really matter what you show, but don't change the data type.
   b. For EmailAddress, we should see the first two letters, then ".abc.def@xyz.com".
   c. For PhoneNumber, show first two letters, then 1-111-11, then last two letters. It is a string datatype.
   d. For CardNumber, keep the proper formatting, but first three groups should be ****, as in **** **** **** 1234.
6. Use T-SQL for above, there's no GUI in SSMS for Dynamic Data Masking.
7. Grant Lisa permissions to see unmasked data.
8. Use EXECUTE AS USER = 'username' to test both users. Use REVERT to "be yourself" again.
9. Remove the masking for all columns.

Ex 4: Always Encrypted

This is a more advanced lab. Do it if you feel like it and have time for it.

1. You will encrypt the CardNumber column in the Persons table used above.
2. Randomized encryption.
3. You can use the Wizard in SSMS, makes it very easy.
4. Or use T-SQL, easiest is to make a copy of this table, like Persons2.
5. If you used the wizard, SELECT from the table. You should see the column encrypted.
6. Use SSMS to connect with proper connection and parameterization option, string and try SELECT again.
7. If you feel advanced, try an INSERT or an UPDATE, using variables in SSMS.

Lab 3 answer suggestions

Ex 1. Containment

```sql
USE master
GO

--Create a database to play with
DROP DATABASE IF EXISTS myDatabase
CREATE DATABASE myDatabase
GO

--Allow containment at the instance level
EXEC sp_configure 'contained database authentication', 1
RECONFIGURE

--Kick out existing users
ALTER DATABASE myDatabase SET SINGLE_USER WITH ROLLBACK IMMEDIATE
WAITFOR DELAY '00:00:02'
ALTER DATABASE myDatabase SET MULTI_USER WITH ROLLBACK IMMEDIATE

--Set database containment to partial
ALTER DATABASE myDatabase SET CONTAINMENT = PARTIAL

--Create user with password
USE myDatabase

CREATE USER myUser WITH PASSWORD = 'myPassword'

--Connect using Object Explorer
--Set above database as the database to connect to

--Right-click the database, and New Query
--In that Query window, try:
USE model
--Close this connection

--Above should fail since guest isn't enabled in model

--Cleanup
USE master
DROP DATABASE myDatabase
```

Ex 2. Data classification and vulnerability report

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

Ex 3. Dynamic Data Masking

```
--Dynamic data Masking

USE Adventureworks

--Create users to play with
DROP USER IF EXISTS Kalle
DROP USER IF EXISTS Lisa
CREATE USER Kalle WITHOUT LOGIN
CREATE USER Lisa WITHOUT LOGIN

--Grant permissions
GRANT SELECT ON Persons TO Kalle, Lisa

SELECT * FROM Persons

--Turn on Dynamic Data Masking for the desired columns
ALTER TABLE Persons ALTER COLUMN BirthDate ADD MASKED WITH (FUNCTION = 'default()')
ALTER TABLE Persons ALTER COLUMN EmailAddress ADD MASKED WITH (FUNCTION = 'partial(2,
".abc@def.com", 0)')
ALTER TABLE Persons ALTER COLUMN PhoneNumber ADD MASKED WITH (FUNCTION = 'partial(2,
"1-111-11", 2)')
ALTER TABLE Persons ALTER COLUMN CardNumber ADD MASKED WITH (FUNCTION = 'partial(0,
"**** **** **** ", 4)')

--Lisa should see unmasked data
GRANT UNMASK TO Lisa

--Test both Lisa and Kalle
EXECUTE AS USER = 'Lisa'
SELECT * FROM Persons
REVERT

EXECUTE AS USER = 'Kalle'
SELECT * FROM Persons
REVERT


--Cleanup
--Remove the masks
ALTER TABLE Persons ALTER COLUMN BirthDate DROP MASKED
ALTER TABLE Persons ALTER COLUMN EmailAddress DROP MASKED
ALTER TABLE Persons ALTER COLUMN PhoneNumber DROP MASKED
ALTER TABLE Persons ALTER COLUMN CardNumber DROP MASKED

DROP USER IF EXISTS Kalle
DROP USER IF EXISTS Lisa
```

Ex 4: Always Encrypted

Lab answer suggestions in the "Solution Always Encrypted.sql" file.

Lab 4: Transact-SQL enhancements

Do whichever exercise(s) you feel for and in whatever order you want.

Preparation

Run the **PrepareLabDML.sql** file.

Ex 1. Sequence

1. Create a sequence start value 1, increment 1 data type bigint. Use T-SQL directly or the GUI.
2. Create a table with an int as PK.
3. Insert a few rows into the table, and fetch value for the PK from the sequence.
4. Modify the table (or create a new table), add a default for the PK column using the sequence.
5. Insert a few more rows.

Ex 2. Manage DDL

1. You have an install script for tables, old-school, named InstallMyDatabaseLabTables.sql.
2. Do not add error handling such as TRY/CATCH, that isn't the purpose for the lab.
3. It assumes a database named myDatabaseLab.
4. Execute the script. It should fail since the database doesn't exist.
5. Create the database.
6. Execute it, see that it runs.
7. Execute it again, see that it doesn't fail.
8. Change it to utilize improvements in more recent versions of SQL Server. I.e., dropping things if they exist.
9. Execute it, see that it runs.
10. Execute it again, see that it doesn't fail.
11. Now modernize the script file that creates views and stored procedures: InstallMyDatabaseLabCode.sql
12. We don't want to lose permissions assigned to the views and stored procedures.

Ex 3. Handling dirty data (using TRY_CAST)

1. You have a table called DirtyPersons, with dirty data.
2. Your will import this data into the CleanPersons table.
3. Investigate both tables regarding data types (sp_help /Alt-F1 is useful here).
4. The DirtyPersons table is a staging table, data was imported from a text file. There are strange data in the BirthDate and Age columns.
5. Your job is to get this data into the CleanPersons Table. For rows that aren't valid values, we want NULL.
6. We also want the original value in the "Imported" columns for the rows/columns that are dirty.
7. Don't worry that the Age doesn't match the BirthDate, that is not intended and fixing that is not part of the exercise.
8. TRY_CAST or TRY_CONVERT can be useful here. (Or to some extent even TRY_PARSE, if you are into that.)
9. Now investigate the CleanPersons table, optionally SELECT only the problematic rows.

Ex 4. Other useful functions

1. Use STRING_SPLIT to get a table out of this comma separated string 'green, blue, white, red'
2. You have this weird SalesComma table with comma-separated product numbers. Create a query so you get one row per combo of SalesOrderID and Product. Tip: Both STRING_SPLIT and CROSS APPLY will be useful.
3. Store the result from above in a table named mySales, using SELECT INTO
4. Now use mySales to create a table with same structure as SalesComma. No cheating, the comma-separated Products column should have products sorted by product number.
5. Use the view HumanResources.vEmployee to return two columns. One named "who" in which you have businessenityid, first-, middle-, and lastname separated bu a space. And another column named "where" in which you have city, state and country separated by comma and space.

Ex 5. Temporal

1. You can either convert an existing table to a temporal table, or create a new one.
2. Use the PersonsAge table.
   a. Either create a new with same structure, but with temporal support, named myPersonsAge.
   b. Or convert the existing PersonsAge to temporal.
   c. The temporal columns should be visible.
   d. The history table has the same name as the temporal table, but ends with "History".
   e. You want a retention period of 12 months.
3. Now play with them, try to INSERT, UPDATE and DELETE some data.
4. Check out the history table.
5. Try the FOR SYSTEM TIME AS OF clause.

Lab 4 answer suggestions

Ex 1. Sequence

Lab answer suggestions in the "Solution Sequence.sql" file.

Ex 2. Manage DDL

Lab answer suggestions in the "Solution InstallMyDatabaseLabTables.sql" and "Solution InstallMyDatabaseLabObject.sql"files.

Ex 3. Handling dirty data (using TRY_CAST)

Lab answer suggestions in the "Solution Handling dirty data.sql" file.

Ex 4. Other useful functions

Lab answer suggestions in the "Solution Other useful functions.sql" file.

Ex 5. Temporal

Lab answer suggestions in the "Solution Temporal.sql" file.

Lab 5: Indexes

Note: Lab will use AdventureworksDW, unless noted differently.

Do whichever exercise(s) you feel for and in whatever order you want.

Preparation:

Run the PrepareLabIndex.sql file.

Ex 1. New index syntax and options

1. You will create a table, and try some new index syntax and functionality.
2. The name of the table should be Personer.
3. The columns should be PersonNummer, FörNamn and EfterNamn.
4. Details for nullability and data types aren't important.
5. PersonNummer should be PK and have a clustered index.
6. The other columns should have nonclustered indexes.
7. Efternamn should be page compressed and förnamn row compressed.
8. Everything should be specified in the CREATE TABLE command. No CREATE INDEX.

Ex 2. Convert rowstore to Columnstore

1. You have a table named FactResellerSalesXL, which is a rowstore table.
2. It has a B-Tree clustered index.
3. Convert the table to a columnstore table. I.e., the table should have a clustered columnstore table. Use only one command to do this.

Ex 3. Study alignment

1. You have this FactResellerSalesXL_Copy table, with a clustered columnstore index.
2. If not, create it using the solution for exercise 2.
3. We often search by OrderDateKey, which is an int
4. Use my demo files (or some other means) to find out:
5. How many segments are there?
6. Do you have good alignment based on OrderDateKey?

Ex 4. Fix alignment

1. You have this FactResellerSalesXL_Copy table, with a clustered columnstore index.
    a. If not, create it using the solution for exercise 2.
2. The alignment sucks, for your usage pattern (OrderDateKey).
    a. Fix it.
3. Verify the alignment.

Ex 5. Resumable index rebuild

1. You have this FactResellerSalesXL table, with a clustered row-store index.
2. Set recovery for the database to full.
3. Backup the database to the file nul.
4. Backup the log to the file nul.
5. Perform a rebuild, make it resumable.
6. Be prepared to pause the operation, from a different query window after some 20 seconds.
7. Stop the operation after some 20 seconds.
8. Check how many MB you have in the transaction log
9. Empty the transaction log (backup to nul).
10. Check how many MB you have in the transaction log.
11. Check how far you are with the index rebuild.
12. Resume the operation.
13. Check how many MB you have in the transaction log.

Lab 5 answer suggestions

Ex 1. New index syntax and options

Lab answer suggestions in the "Solution New index syntax and options.sql" file.

Ex 2. Convert rowstore to Columnstore

Lab answer suggestions in the "Solution Convert rowstore to columnstore.sql" file.

Ex 3. Study alignment

Lab answer suggestions in the "Solution Study alignment" file.

Ex 4. Fix alignment

Lab answer suggestions in the "Solution Fix alignment.sql" file.

Ex 5. Resumable index rebuild

Lab answer suggestions in the "Solution Resumable index rebuild.sql" file.

Lab 6: Performance

Do whichever exercise(s) you feel for and in whatever order you want.

Note: Labs will use AdventureworksDW, unless noted differently.

Preparation:

Run the PrepareLabPerformance.sql file.

Ex 1. Turn on Query Store

1. Turn on the query store.
2. For the sake of the lab:
    a. Write to the database every 60 seconds
    b. Aggregate every minute
    c. Capture all queries, not just "expensive ones"
    d. Allow for 1 GB of Query Store Data
3. Above are not best practices for a production environment!

Ex 2. Fix slow query 1

1. You have a slow query, in the "Fix slow query 1.sql" file.
2. Make it quicker.
3. Changing the query or table structure is not an option.
4. Similar queries should also benefit from your fix.

Ex 3. Fix slow query 2

1. You have a slow query, in the "Fix slow query 2.sql" file.
2. Make it quicker.
3. You are allowed to modify the query.
4. Your fix should not affect other queries in this database.

Ex 4. Lightweight profiling

1. Use the file "Lightweight profiling.sql".
2. Set compat level to 130 (we need a slow query).
3. Turn on last query plan stats. It is a database scoped configuration, not exposed in the GUI.
4. Turn on live exec plan and run the slow query, see if you can visually identify what is taking longest.
5. Now you want to find the actual plan, from the DMVs, for this query. Depending on your experience, use any of below:
    a. Less experienced:
        i. You have been given the query hash
        ii. Grab the plan_handle for this (sys.dm_exec_query_stats)
        iii. Grab the actual plan for this plan_handle (sys.dm_exec_query_plan_stats)
    b. More experienced:
        i. Do it all in one query

Ex 5. Look at Query Store (requires earlier exercises)

1. You have had Query Store running since Ex 1.
2. Use the GUI to check out the reports for Query Store.
3. For example, which is the most expensive query.

Lab 6 answer suggestions

Ex 1. Turn on Query Store

Lab answer suggestions in the "Solution Turn on Query Store.sql" file.

Ex 2. Fix slow query 1

Lab answer suggestions in the "Solution Fix slow query 1.sql" file.

Ex 3. Fix slow query 2

Lab answer suggestions in the "Solution Fix slow query 2.sql" file.

Ex 4. Lightweight profiling

Lab answer suggestions in the "Solution Lightweight profiling.sql" file.

Ex 5. Look at Query Store (requires earlier exercises)

No lab answer for this. Use the lab instructions.

Lab 7: Management

Ex 1. Extended Events

1. You want to capture is strange stuff happens on your SQL Server.
2. Use Tibor's "Looking for strange" trace.
3. To generate "load", click around in Object Explorer.
4. See how many of the events you captured.
5. Optional: reconfigure the trace to also capture the actual event info to a file.
    a. Only for the events you find interesting from above.
    b. Re-generate your load.
    c. Check the file target for the events in question.

Lab 7 answer suggestions

No lab answer for this. Use the lab instructions.