# Week 12 Lab 1 Databases 3 Practical:

# User Accounts and Privileges 1 of 2 (total 10%)

## Creating a new database

## Note – These tasks are to be completed on your database server running in a container under Docker.

## You have been given the task of creating a database for a car sales yard.

1. Open SQL Server management studio, connecting as SA to your containerized database.
2. On the object explorer panel, right click on "Databases" and select "New Database". Maximize the dialog box
3. In the database name field, type Cars. Make sure to set the owner "sa"
4. Under the database files section, take note of the columns and their values

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Logical Name | File Type | Filegroup | Initial Size (MB) | Autogrowth / Maxsize | Path | File Name |
| Cars | ROWS Data | PRIMARY | 8 | By 64 MB, Unlimited | /var/opt/mssql/data/ |  |
| Cars\_log | LOG | Not Applicable | 8 | By 64 MB, Unlimited | /var/opt/mssql/data/ |  |

1. Enter "Cars.mdf" for the default primary data file's file name
2. Enter "Cars.ldf" for the default primary log file's file name.
3. The "Logical Name" is different to the "File Name". Describe what each one means. The logical name **uniquely identifies a file within a database for use in commands like ALTER DATABASE**

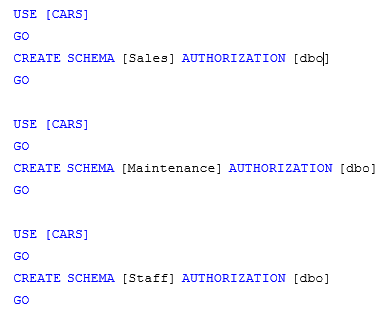
Whereas the File Name is **the physical name refers to the path to the file on disk.**

1. Click the script button at the top, and then check the query window to see the generated script. You can cancel out of the database creation dialog window.
2. A query window will have opened showing the scripts that will be run to create the Cars database with the options we just specified. Familiarize yourself with this process. You will be expected to create a databases for multiple users via a script.
3. Select "Master" as the database to execute the query on and click execute.
4. Close the query and refresh the database list.

## Creating new schemas and tables

## We are going to create new schemas to demonstrate their usefulness. We will be applying user permissions to schemas to allow fine-tuned access to any tables contained in the schema.

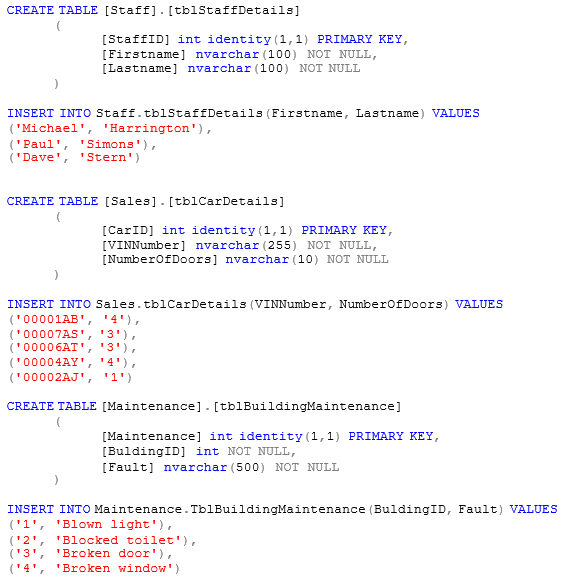
1. Create three new schemas from the code below



1. What is meant by the authorization clause?

**Owner of the schema**

1. Now we have three new schemas, we will create some tables



# Creating new logins and database users

## To allow humans to access the SQL Server database engine they first need a SQL Server login, which is created by the create login command. To then access a database they need a database user account which is mapped to their SQL Server login.

## Expand the security object from the tree in the object explorer

## Expand logins

1. This is the GUI container for all SQL Server logins that are present on the server. Where is DBMS getting this information from?

In SQL Server, there is a catalog view (ie: system view) called *sys.sql\_logins*. This system view that returns all the Logins that have been created in SQL Server as well as information about these Logins.

To retrieve all the logins we can run the following query.

**SELECT \* FROM sys.sql\_logins**

1. There are some built in logins that are required for SQL Server to function correctly, list the login that is installed by default and cannot be removed.

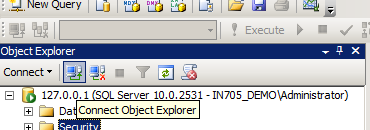
🡪BUILTIN\Administrators

🡪NT AUTHORITY\NETWORK SERVICE

🡪 NT AUTHORITY\SYSTEM

* sa

1. Right click on the logins container and select "New Login"
2. We want to create a new login called Mary. Select SQL Server authentication
3. Enter "ComplexP@ssw0rd" for the password.
4. Un‐tick "User must change password at next login".
5. Click the script button and click cancel.
6. Make sure you understand this command syntax
7. Execute this script
8. Change it so that it will create a login for the username "Mike" with the password "veryComplexP@ssw0rd"
9. Using the object explorer connect tool, open a connection to your SQL Server using Mary's account.



## Your object explorer should look like the following

## Using the Mary connection, expand databases then cars

1. Were you able to? **Not accessible.**

Why?

## Why?

**We have not given permission to that user**

## Under your first connection (this should be your administrative account) expand databases then expand cars

## Expand security then users, right click on users and select "New User"

## Enter Mary for the username and the login name. Enter "dbo" for the default schema

At this point, take a look at the available options for database role membership. Do not select any of the check boxes

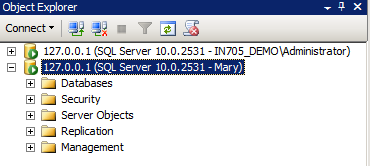
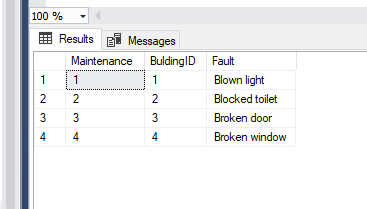
1. Click the script button and click cancel
2. Make sure you understand this command syntax
3. Execute the commands
4. Expand schemas. Where did the maintenance, sales and staff schemas come from?

**From Cars database**

1. Open the Maintenance schema and select permissions
2. Click search and look for the database user Mary. Select this user
3. Give Mary permissions to select **ONLY**. Click the script button and click cancel
4. Make sure you understand this command syntax
5. Execute the commands
6. What does "With Grant" mean?

**It is basically a method for delegating permissions when you have someone you can trust to manage those permissions.**

1. Collapse the connection tree for your administrative connection to the SQL Server and expand the one for the Mary connection. Your object explorer should now look like this (plus a few extra directories).



1. What do you think Mary will be able to access now?

Mary has permission for **ONLY** “SELECT” statement on “Maintenance” schema table “tblBuildingMaintenance”.

For other schemas Mary doesn’t have any permission.

1. Expand databases then try expanding the cars database
2. What tables can you see under the Mary connection?

Maintenance.tblBuildingMaintenance

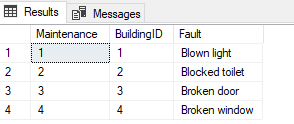
1. Why can't we see the other tables using the Mary connection?

Because we have not set **any** permission for other tables.

1. Right click on the cars database using the Mary connection, and select "New Query"
2. Run SELECT \* FROM Maintenance.tblBuildingMaintenance
3. Did you get rows returned?

Yes

Result :



1. Run DELETE FROM Maintenance.tblBuildingMaintenance
2. What was the result

The DELETE permission was denied on the object 'tblBuildingMaintenance'.

1. Discuss how SQL Server knew not to let the delete command run. Also, how could we have enabled the select permission by default?

Even though we didn't set any deny permissions, by default, if a user is not granted with a permission the default action is to deny .

1. If we wanted to assign select (read) permissions to all the tables in the database, without using schemas, how could we do this?

GRANT SELECT ON DATABASE::[Cars] TO Mary

GO