SQL Server 2008 Tutorial

EGCI 321: Lecture 07 (Week04)

Outline

This tutorial consists of the following lessons:

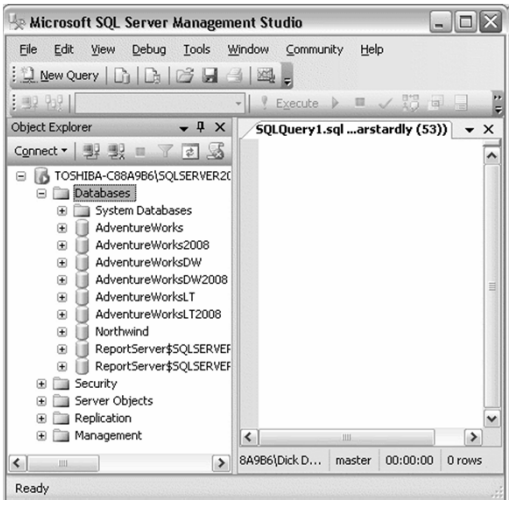
- SQL Server Editions
- SQL Server - Management Studio
- SQL Server - Create a Database
- SQL Server - Create a Table
- SQL Server - Adding Data
- SQL Server - Query Designer
- SQL Server Views
- SQL Server Stored Procedures
- User Logins
- Server Roles
- Database Schemas
- Summary

SQL Server Editions

- Enterprise Edition
  Data management and business intelligence platform providing enterprise class scalability, high availability, and security for running business-critical applications
- Standard Edition
  Data management and business intelligence platform providing ease of use and manageability for running departmental applications
- Workgroup Edition
  Data management and reporting platform providing secure, remote synchronization, and management capabilities for running branch applications
- Developer Edition
  May be installed and used by one user to design, develop, test, and demonstrate your programs on as many systems as needed
- Express Edition
  A free edition of SQL Server ideal for learning and building desktop and small server applications and for redistribution by ISVs
- Compact Edition
  A free, SQL Server embedded database ideal for building stand-alone and occasionally connected applications for mobile devices, desktops, and web clients
- Evaluation Edition
  This edition may be installed for demonstration and evaluation purposes until an expiration period of 180 days.

SQL Server Management Studio (SSMS)

- SSMS enables you to create database objects
  - Such as databases, tables, views
  - View the data within your database
  - Configure user accounts
  - Transfer data between databases

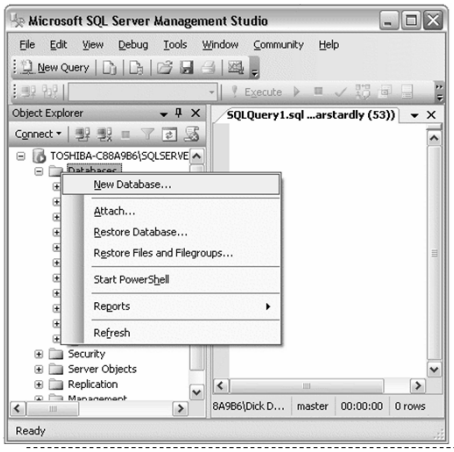


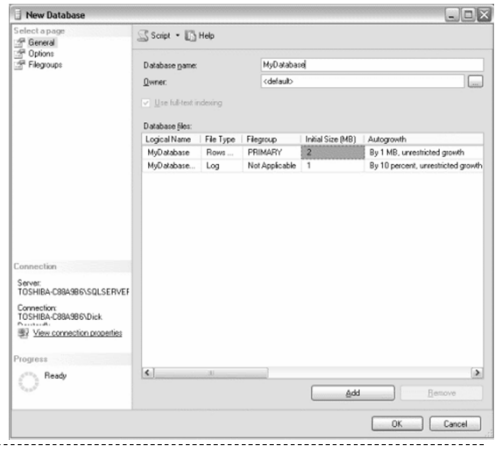
System Databases

Database	Type	Description
master	System database	Stores system level information such as user accounts, configuration settings, and info on all other databases.
model	System database	This database is used as a template for all other databases that are created.
msdb	System database	Used by the SQL Server Agent for configuring alerts and scheduled jobs etc
tempdb	System database	Holds all temporary tables, temporary stored procedures, and any other temporary storage requirements generated by SQL Server.

Creating a New Database

1. Right click on the "Databases" icon and select "New Database..."
2. Name your database and click "OK"



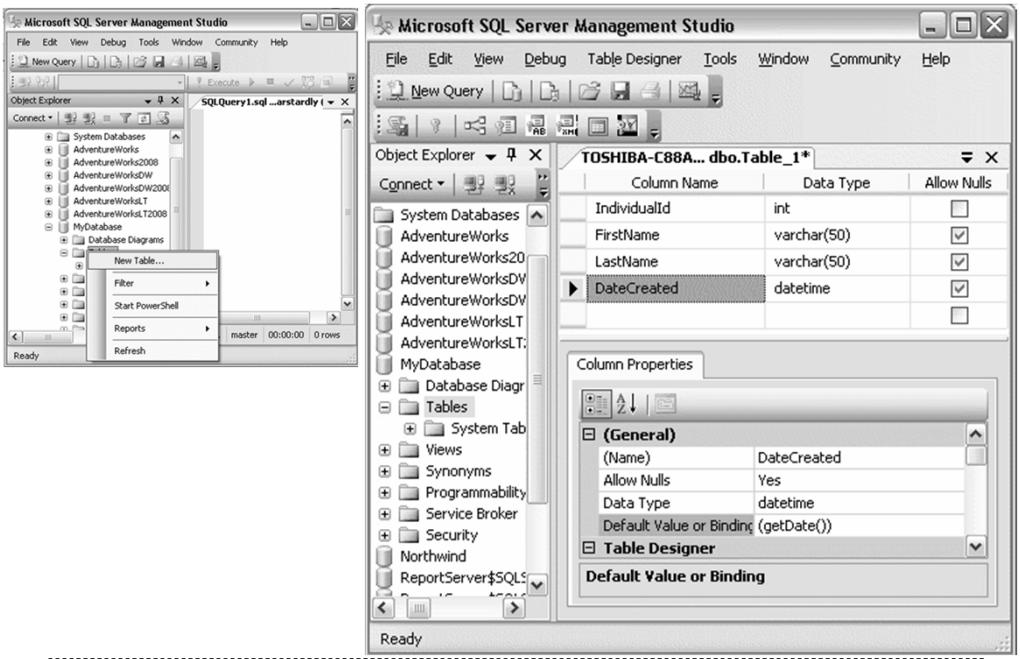


SQL Server - Create a Table

1. Ensuring you have the right database expanded, right click on the "Tables" icon and select "New Table..."
2. While you have this screen open, do the following:
   - Make “IndividualId” column, by setting "Is Identity" to "Yes" (this option is under the "Identity Specification" section in the bottom pane).

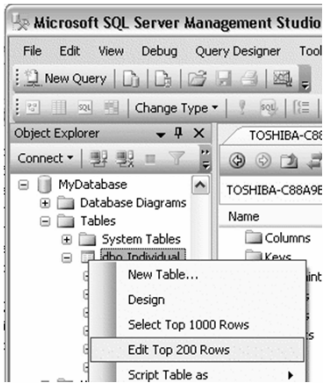
     Note:

     - To set values in the bottom pane, you need to select the column name in the top pane first).
     - This column is going to be an auto-number column - it will contain an incrementing number for each record that is created.
     - Set the "Default Value" of the DateCreated column to *(getdate())*
       - This will automatically insert the current date into that field for each new record.
3. Save the table by selecting *File > Save Table_1*:



Editing Table Rows

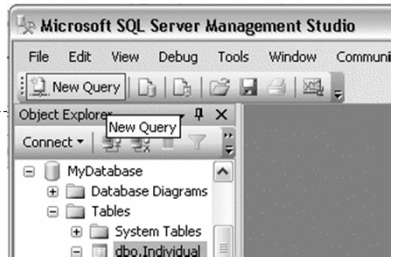
1. To use this option, right click on the table you wish to open, and select "Edit Top 200 Rows":
2. You can now start entering the data directly into your table.



	IndividualId	FirstName	LastName	DateCreated
	1	Homer	Simpson	2009-04-27 11:3...
!	NULL	Barney	Rubble	NULL
!	NULL	Ozzy	Osbourne	NULL
	NULL	Fred	Flinsto	NULL
*	NULL	NULL	NULL	NULL

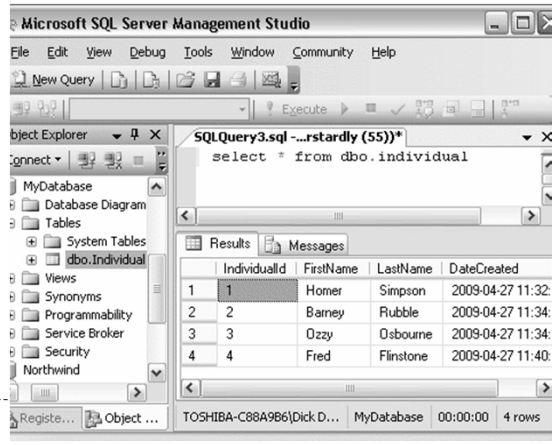
Create a New Query

- Just click the "New Query" button:



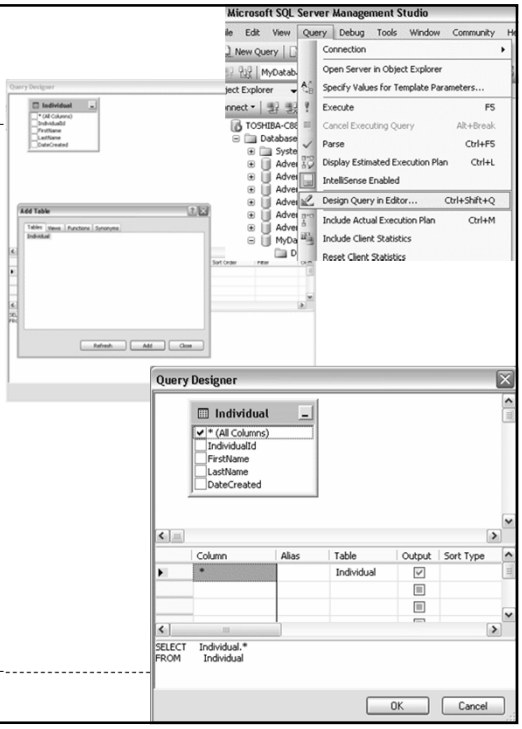
Write/Run Your SQL Script

1. Type your query into the workspace on the right pane
2. Click "Execute" (you can also press F5)



Building Your Queries

1. Select *Query > Design Query in Editor…*
2. Add the tables you want to run the query against. In this case, we only have one table to choose from
3. Select the column/s you want to display in your query



Benefits of Views

- *view* is a pre-written query that is stored on the database.
  - *view* consists of a SELECT statement, and when you run the view
    - you see the results of it like you would when opening a table
  - Some people like to think of a view as a *virtual table*

A view can be useful when there are multiple users with different levels of access, who all need to see portions of the data in the database

- Restrict access to specific *rows* in a table
- Restrict access to specific *columns* in a table
- *Join columns* from multiple tables and present them as though they are part of a single table
- Present *aggregate information* (such as the results of the COUNT function)

Creating a View

```
CREATE VIEW ViewName AS
SELECT ...
```

Ex.

```
CREATE VIEW "Alphabetical list of products" AS
SELECT Products.*, Categories.CategoryName
FROM Categories INNER JOIN Products ON
        Categories.CategoryID = Products.CategoryID
WHERE (((Products.Discontinued)=0))
```


Modifing a View

```
ALTER VIEW "Alphabetical list of products" AS
SELECT Products.*, Categories.CategoryName
FROM Categories INNER JOIN Products ON
        Categories.CategoryID = Products.CategoryID
WHERE (((Products.Discontinued)=0))
```

Running a View

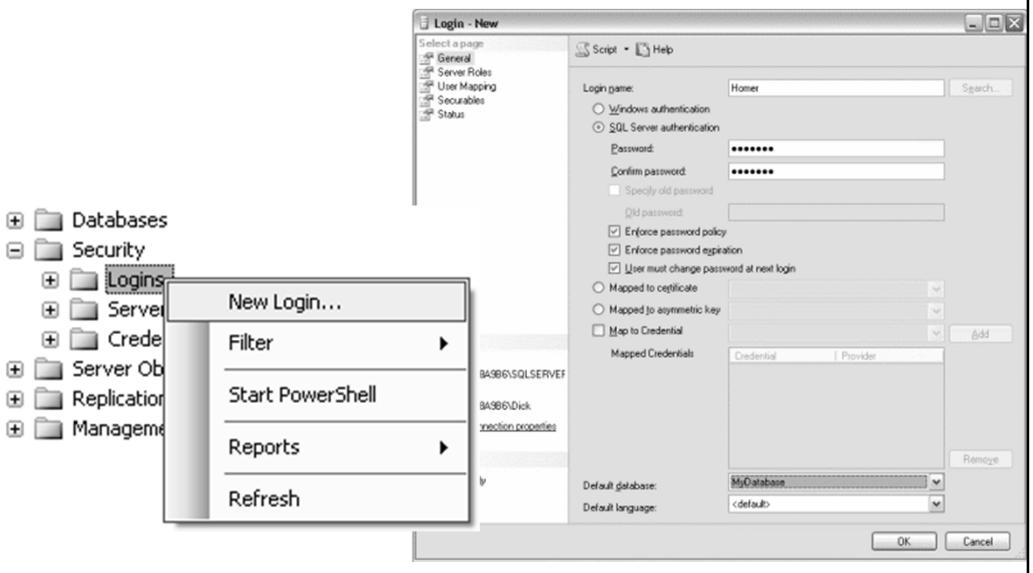
```
SELECT TOP 1000 *
FROM [AdventureWorks2008].[Sales].[vIndividualCustomer]
```

To Create a New User Login

1. Using SQL Server Management Studio, expand the "Security" option and right click on "Logins"
2. Click on “New Login”
3. Complete the login properties in the "General" tab by providing a name for the login, choosing the Authentication method (providing a password if you choose "SQL Server authentication"), and selecting the database to use as a default.
   - If you don't choose a language, it will use the default for the current installation of SQL Server.

Note:

- *If you get an error that reads "The MUST_CHANGE option is not supported by this version of Microsoft Windows", simply uncheck the "User must change password at next login" option.*
- *The error occurs because your operating system doesn't support this option.*

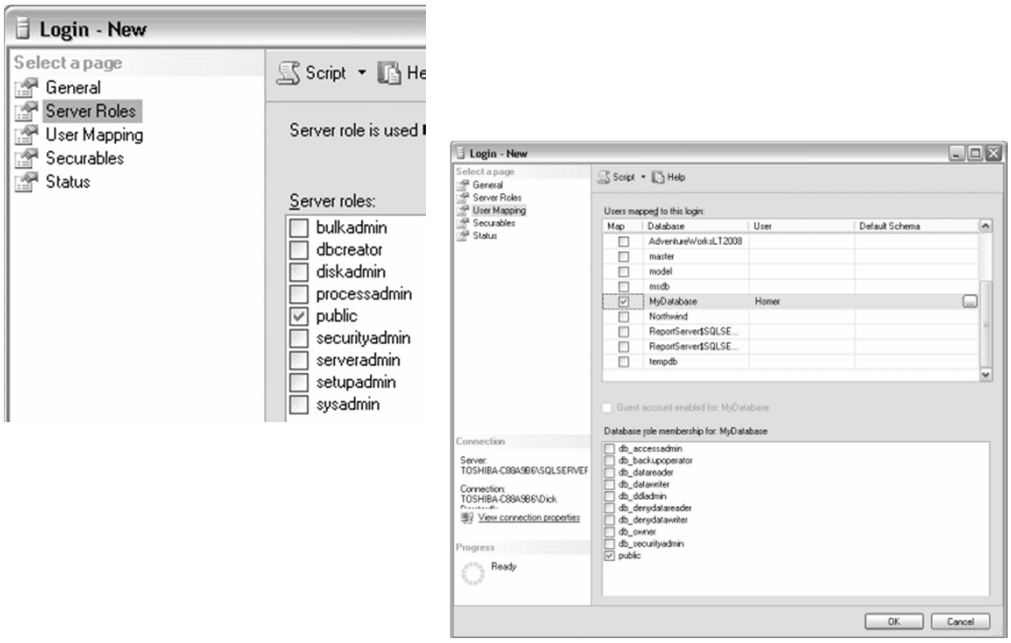


To Create a New User Login (cont.)

4. Click the "Server Roles" tab if you need to apply any server-wide security privileges
5. Click the "User Mapping" tab to specify which databases this user account is allowed to access
   - By default, the login will be assigned to the "Public" role, which provides the login with basic access
   - If the login needs more access in one or more databases, it can be assigned to another role with greater privileges

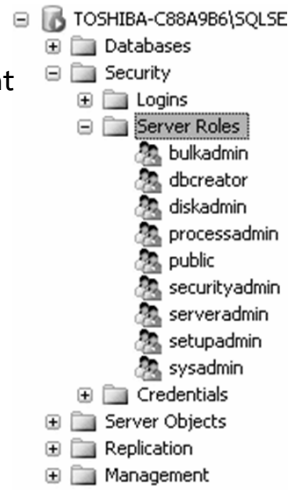
Note:

- *These roles are "Database Roles" and are different to the server roles in the previous tab.*
- *Server roles are for administering the SQL Server*
- *Database roles are created within each database and specify what the login can do within that database*



Accessing the Server Roles

- Accessing the Server Roles
- You view the properties of a server role by right clicking on it
- You can then add users to the server role by clicking Add



Explanation of Server Roles

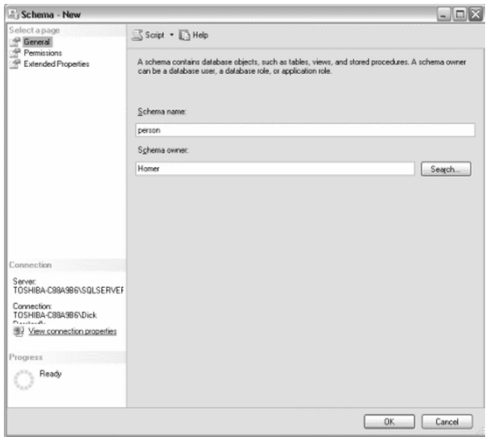
Server Role	Description
sysadmin	Can perform any task in SQL Server.
serveradmin	Can set server-wide configuration options, can shut down the server.
setupadmin	Can manage linked servers and startup procedures.
securityadmin	Can manage logins and database permissions, read logs, change passwords.
processadmin	Can manage processes running in SQL Server.
dbcreator	Can create, alter, and drop databases.
diskadmin	Can manage disk files.
bulkadmin	Can execute BULK INSERT statements.
public	Every SQL Server user account belongs to this server role. When a server principal has not been granted or denied specific permissions on a securable object, the user inherits the permissions granted to public on that object. Only assign public permissions on an object when you want the object to be available to all users.

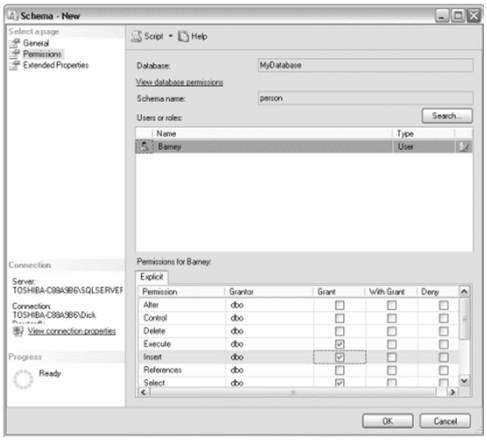
Database Schemas

- A database schema is a way to logically group objects
  - Such as tables, views, stored procedures
  - Think of a schema as a container of objects
- You can assign a user login permissions to a single schema
  - User can only access the objects they are authorized to access
- Schemas can be created and altered in a database, and users can be granted access to a schema
  - A schema can be owned by any user, and schema ownership is transferable

Creating a Database Schema

1. Navigate to *Security > Schemas*
2. Right click on *Schemas* and select *New Schema...*
3. Complete the details in the *General* tab for the new schema
   - In this example, the schema name is "person" and the schema owner is "Homer"
4. Add users to the schema as required and set their permissions
5. Add any extended properties (via the *Extended Properties tab*) Click *OK*

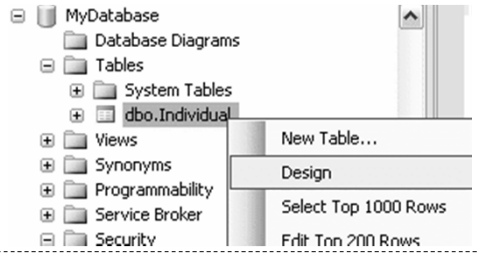




Add a Table to the New Schema

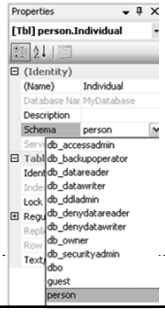
- When we created that table (called "Individual"), it was created in the default database schema ("dbo")
- We know this because it appears in our object browser as "dbo.Individual"
- To transfer the "Individual" table to the person "schema"

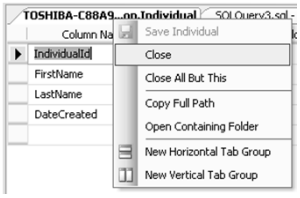
1. In Object Explorer, right click on the table name and select "Design"



Add a Table to the New Schema (cont.)

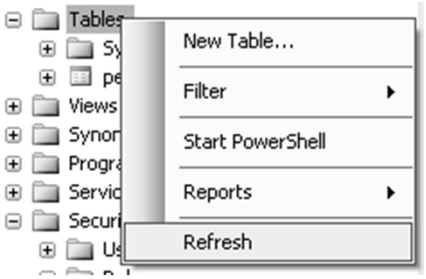
2. From Design view, press F4 to display the Properties window
3. From the Properties window, change the schema to the desired schema
4. Close Design View by right clicking the tab and selecting "Close"
5. Click "OK" when prompted to save

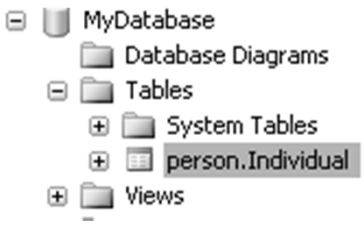




Confirm your Change

1. Refresh the Object Browser view
2. You will now see that Object Browser displays the new schema for the table (person.Individual)





Reference

1. http://www.quackit.com/sql_server/tutorial/