

Databases in VS & Azure

In Cert 4 Database You Learned:

Database Concepts:

RDBMS, Tables, Columns, Rows, Primary Keys, Foreign Keys, Constraints, Indexes

SQL

Queries, DDL, DML, CRUD, Aggregates,

ERD's

Entities, Attributes, Relationships, Cardinality/Participation, Identifiers,

Strong vs weak entities

Relational Schemas

Normalisation – 0NF, 1NF, 2NF, 3NF

Technology

MS Access

SQL Server (Azure) – via Query Editor

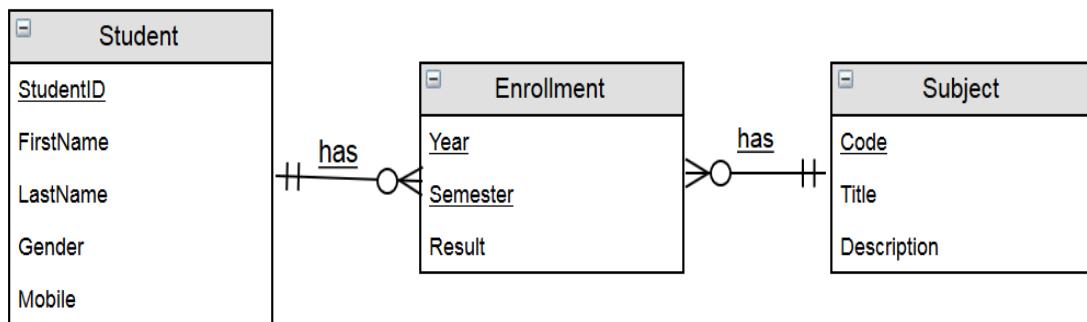
Now it is time to get more ‘practical’

- Building a Database in VS
- Deployment (directly) from VS to Azure
- Populating your database with sample test data (on deployment)
- Making (controlled) changes to your database

(And In conjunction with this weeks lecture)

- Automagically generating a Web API (model & controllers) from DB
- Deploying your Web API to Azure
- Testing Web API Endpoints

1. (Analyse &) Design Your DB etc.



StudentID	FirstName	LastName	Gender	Mobile
s12345678	Michael	Hutchence	M	0400 555 123
s23456789	Kurt	Cobain	M	0411 555 321
s34567890	Amy	Winehouse	F	0422 555 432
s98765432	Janis	Joplin	F	0433 555 543

Code	Title	Description
ICTPRG501	Advanced OO Programming	Do fancy stuff like polymorphism, decoupling, generics, delegates etc.
ICTDBS501	Advanced Database	Do even fancier stuff like TSQL, transactions & ACID, Triggers etc.

STUDENT (STUDENTID, FIRSTNAME, LASTNAME, GENDER, MOBILE)

SUBJECT (CODE, TITLE, DESCRIPTION)

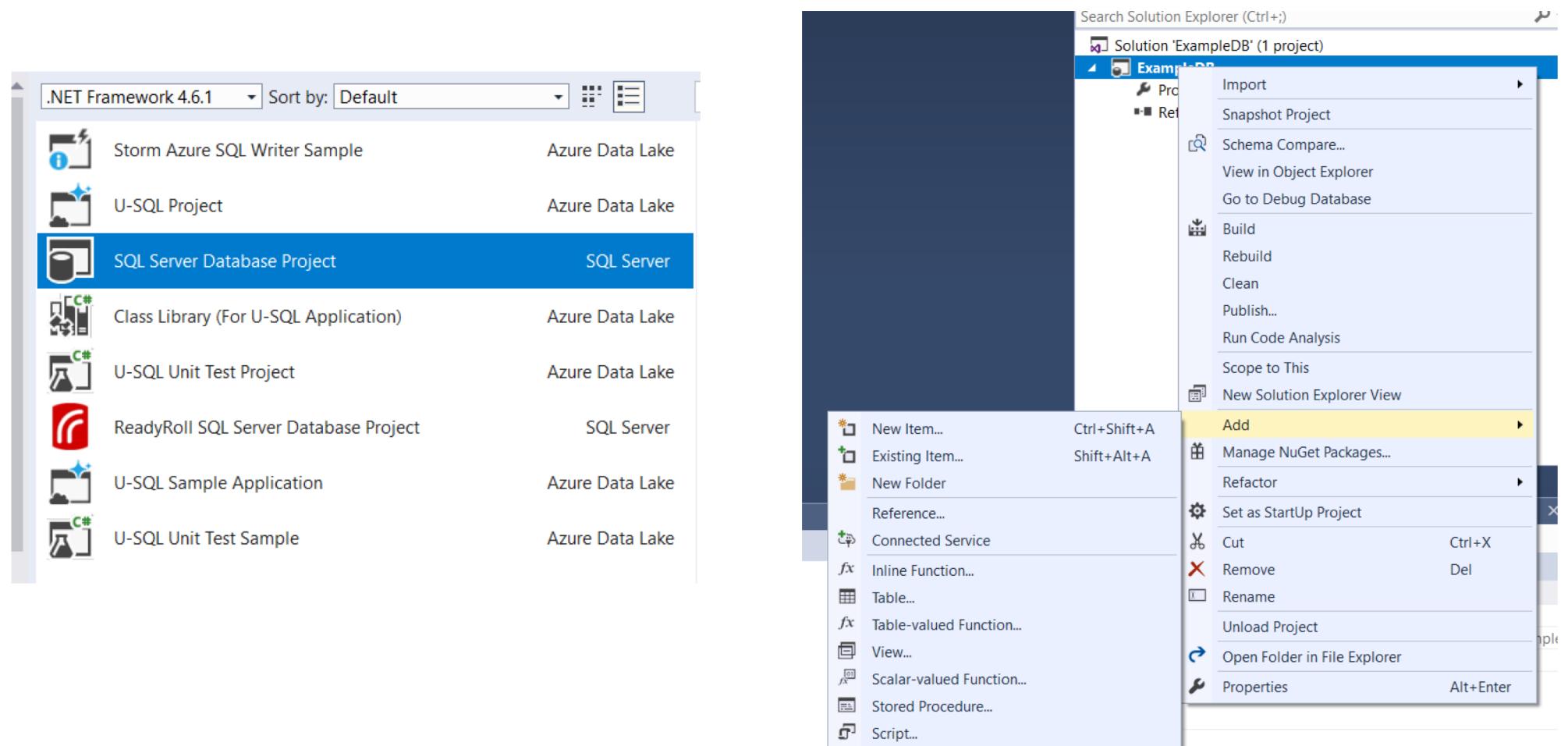
ENROLLMENT (STUDENTID, CODE, YEAR, SEMESTER, RESULT)

FOREIGN KEY STUDENTID REFERENCES STUDENT STUDENTID

FOREIGN KEY CODE REFERENCES SUBJECT CODE

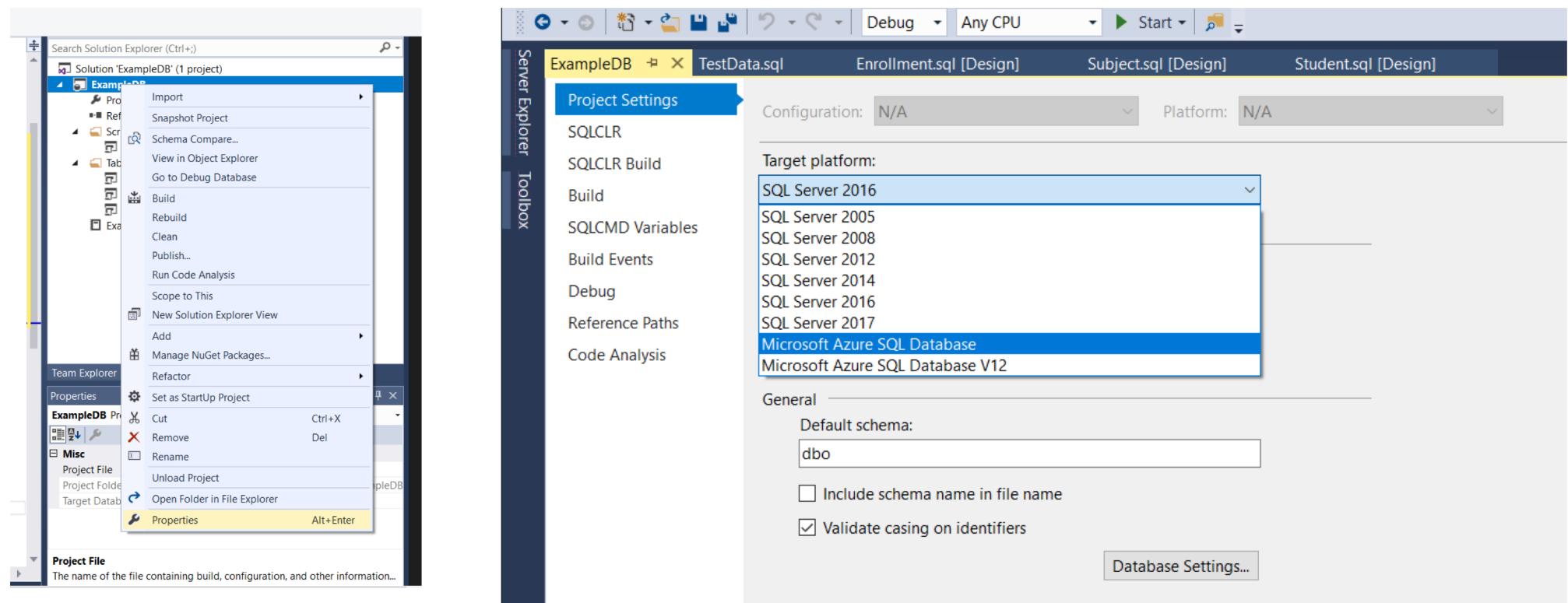
StudentID	Code	Year	Semester	Result
s12345678	ICTPRG501	1996	2	D
s12345678	ICTDBS501	1997	1	SEN
s23456789	ICTDBS501	1994	1	SEN
s98765432	ICTPRG501	1969	2	N
s98765432	ICTPRG501	1970	1	C
s98765432	ICTDBS501	1970	2	SEN
s34567890	ICTDBS501	2011	1	SEN

2. Create a DB Project in VS & Add Tables



Make it an Azure Database

Project Properties -> Project Settings -> Target Platform : Microsoft Azure SQL Database



3. Build Your Tables in SQL (or GUI for the ‘weak’)

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface with the following details:

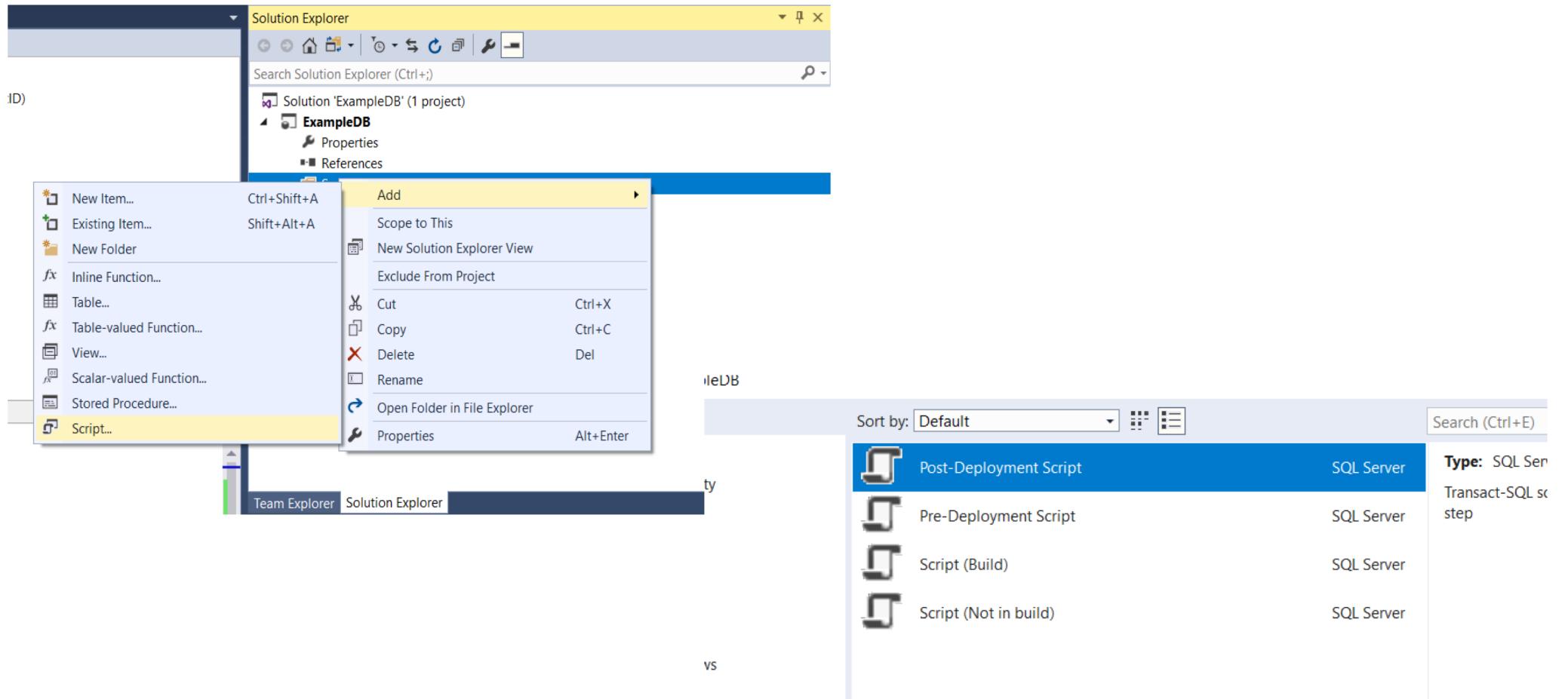
- Solution Explorer:** Shows a solution named 'ExampleDB' with one project. Under 'Tables', there are three files: Enrollment.sql, Student.sql, and Subject.sql, all of which are currently selected.
- Server Explorer:** Shows the connection to 'ExampleDB'.
- Toolbox:** Available on the left side of the interface.
- Script File:** Enrollment.sql
- Design View:** Shows the T-SQL code for creating the 'Student' table:

```
1 CREATE TABLE [dbo].[Student]
2 (
3     [StudentID] NVARCHAR(10) NOT NULL,
4     [FirstName] NVARCHAR(50) NOT NULL,
5     [LastName] NVARCHAR(50) NOT NULL,
6     [Gender] NVARCHAR(1) NULL,
7     [Mobile] INT NULL,
8     CONSTRAINT PK_STUDENT PRIMARY KEY (StudentID),
9     CONSTRAINT CHK_GENDER CHECK (Gender IN ('M', 'F'))
10 )
```
- Design View:** Shows the T-SQL code for creating the 'Subject' table:

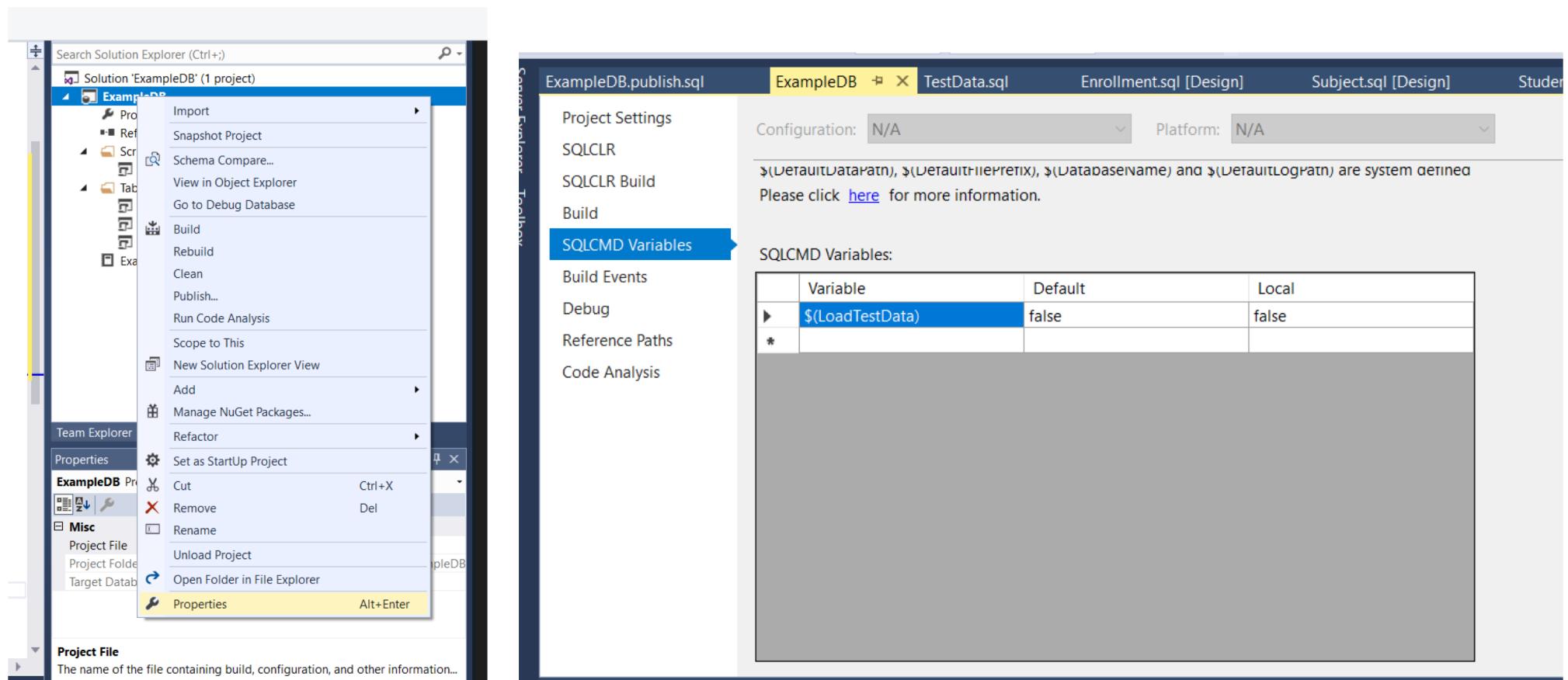
```
1 CREATE TABLE [dbo].[Subject]
2 (
3     [Code] NVARCHAR(10) NOT NULL,
4     [Title] NVARCHAR(50) NOT NULL,
5     [Description] NVARCHAR(200) NULL,
6     CONSTRAINT PK_SUBJECT PRIMARY KEY (Code)
7 )
```
- Design View:** Shows the T-SQL code for creating the 'Enrollment' table:

```
1 CREATE TABLE [dbo].[Enrollment]
2 (
3     [StudentID] NVARCHAR(10) NOT NULL,
4     [Code] NVARCHAR(10) NOT NULL,
5     [Year] INT NOT NULL,
6     [Semester] INT NOT NULL,
7     [Result] NVARCHAR(3) NULL,
8     CONSTRAINT PK_ENROLLMENT PRIMARY KEY (StudentID, Code, Year, Semester),
9     CONSTRAINT FK_ENROL_STUDENT FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
10    CONSTRAINT FK_ENROL SUBJECT FOREIGN KEY (Code) REFERENCES dbo.Subject(Code),
11    CONSTRAINT CHK_RESULT CHECK (Result IN ('HD', 'D', 'C', 'P', 'N', 'SEN', 'WD'))
12 )
```
- Object Explorer:** Shows the structure of the 'Enrollment' table, including its columns, keys, check constraints, foreign keys, indexes, and triggers.

Create a Post Deployment (PD) Script



Set SQLCMD Variable for Loading Test Data



Add Test Data in Post Deployment Script

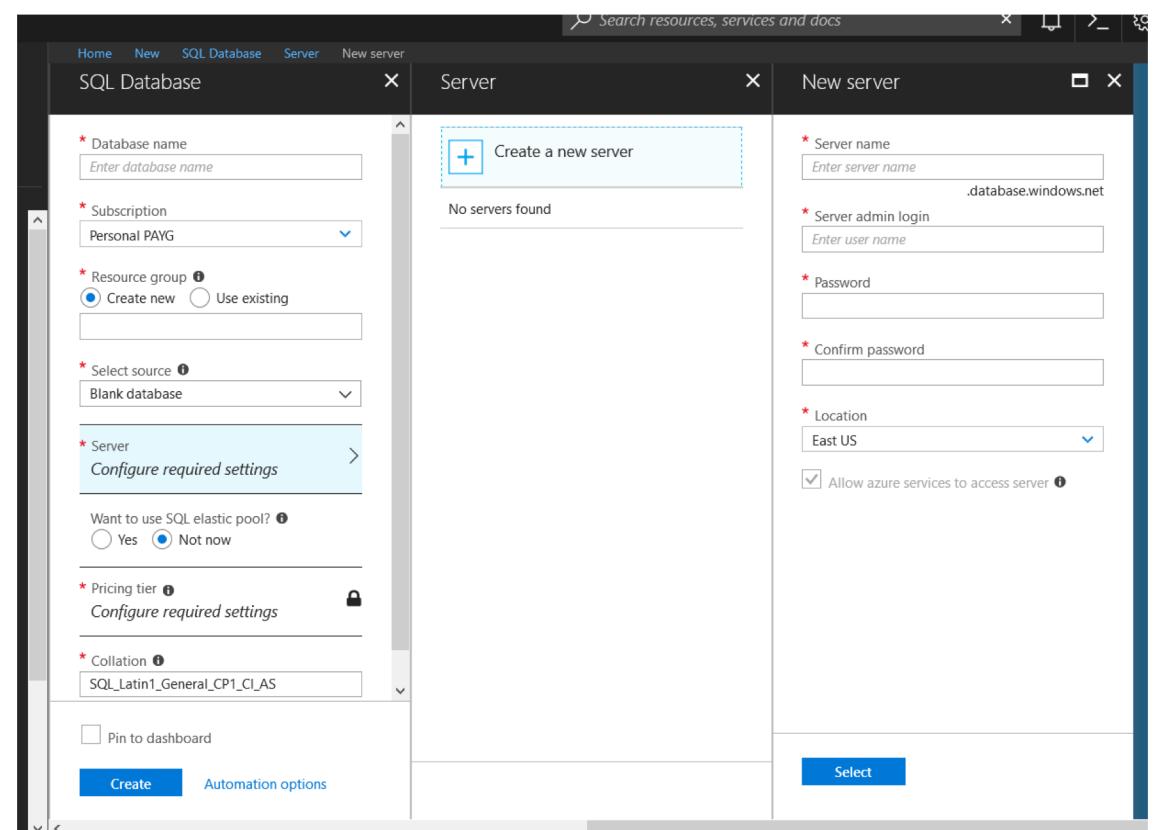
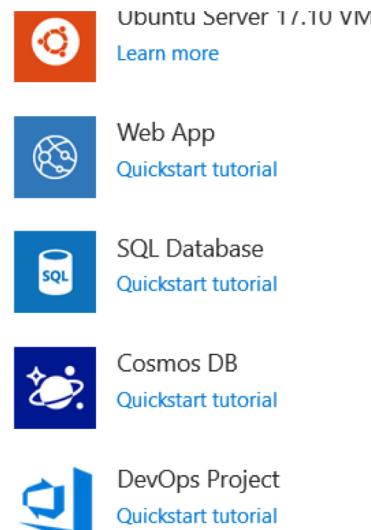
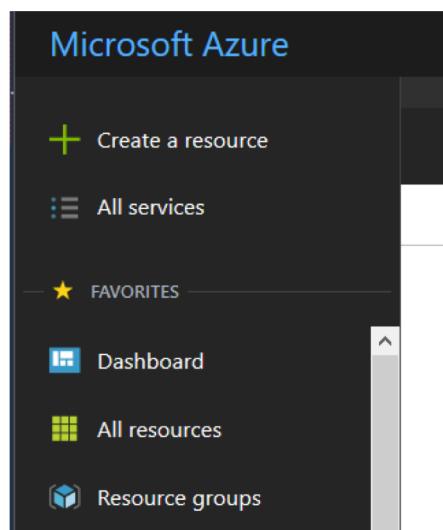
Note the conditional logic
around the SQLCMD variable
\$(LoadTestData)

Also note the BEGIN and END tags

```
11  */
12 IF '$(LoadTestData)' = 'true'
13 BEGIN
14
15 -- ENSURE THERE IS NO DATA IN THE TABLES BEFORE DEPLOYING TEST DATA
16 DELETE FROM ENROLLMENT;
17 DELETE FROM STUDENT;
18 DELETE FROM SUBJECT;
19
20 -- INSERT THE TEST DATA
21
22 INSERT INTO STUDENT (STUDENTID, FIRSTNAME, LASTNAME, GENDER, MOBILE) VALUES
23 ('S12345678', 'Michael', 'Hutchence', 'M', 0400555123),
24 ('s23456789', 'Kurt', 'Cobain', 'M', 0411555321),
25 ('s34567890', 'Amy', 'Winehouse', 'F', 0422555432),
26 ('s98765432', 'Janis', 'Joplin', 'F', 0433555543);
27
28
29 INSERT INTO SUBJECT (CODE, TITLE, DESCRIPTION) VALUES
30 ('ICTPRG501', 'Advanced OO Programming', 'Do fancy stuff like polymorphism, decoupling, generics, delegates etc.'),
31 ('ICTDBS501', 'Advanced Database', 'Do fancier stuff like TSQL, transactions & ACID, Triggers etc.');
32
33
34 INSERT INTO ENROLLMENT (STUDENTID, CODE, YEAR, SEMESTER, RESULT) VALUES
35 ('s12345678', 'ICTPRG501', 1996, 2, 'D'),
36 ('s12345678', 'ICTDBS501', 1997, 1, 'SEN'),
37 ('s23456789', 'ICTDBS501', 1994, 1, 'SEN'),
38 ('s98765432', 'ICTPRG501', 1969, 2, 'N'),
39 ('s98765432', 'ICTPRG501', 1970, 1, 'C'),
40 ('s98765432', 'ICTDBS501', 1970, 2, 'SEN'),
41 ('s34567890', 'ICTDBS501', 2011, 1, 'SEN');
42
43 END;
```

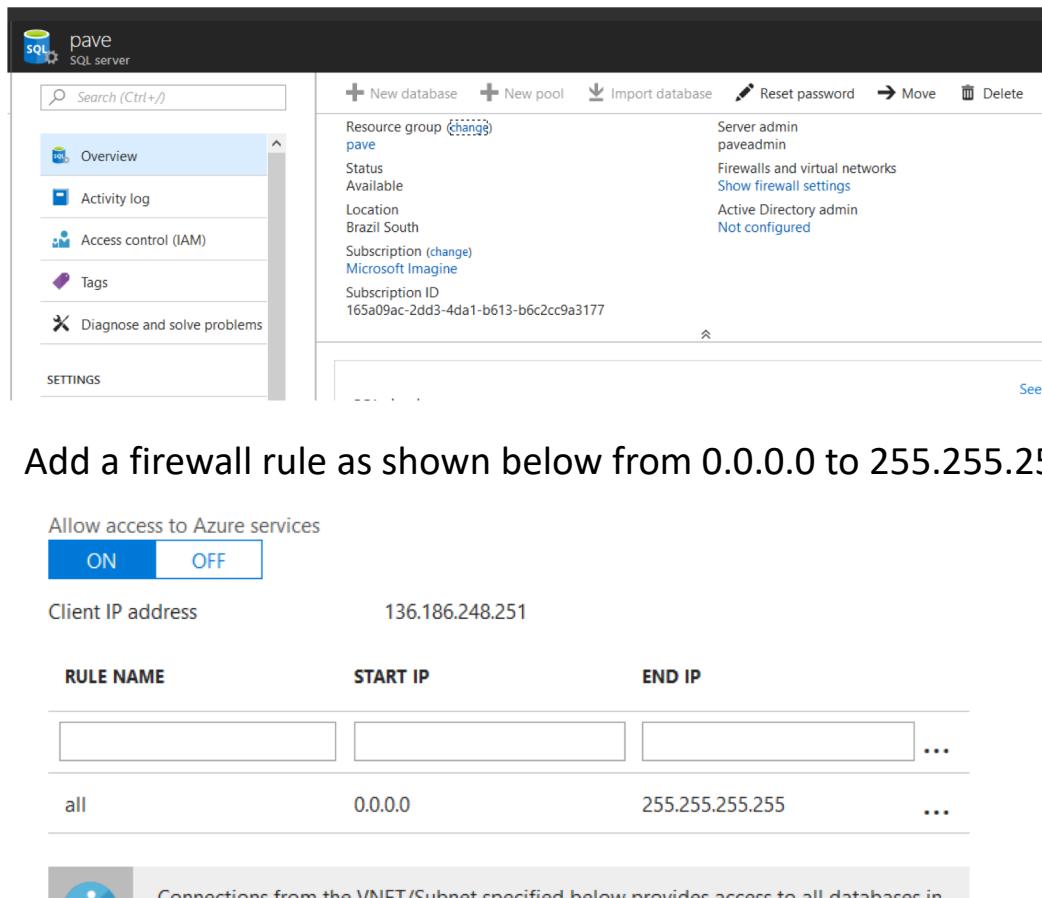
Create DB on Azure

1. Select Create a Resource
2. Select SQL Database
3. Enter the database, & server details
4. Wait for it to deploy



Azure Firewall Rules

On the server settings page select Show Firewall Settings



The screenshot shows the Azure portal interface for a SQL Server named 'pave'. On the left, there's a sidebar with options like Overview, Activity log, Access control (IAM), Tags, and Diagnose and solve problems. The main content area displays basic server details: Resource group (pave), Status (Available), Location (Brazil South), Subscription (Microsoft Imagine), and Subscription ID (165a09ac-2dd3-4da1-b6c2cc9a3177). Below this, under 'Firewalls and virtual networks', there's a link to 'Show firewall settings'. The 'Show firewall settings' section is expanded, showing a toggle switch for 'Allow access to Azure services' which is set to 'ON'. It also lists a single rule: 'Client IP address' is 136.186.248.251. A table shows existing firewall rules:

RULE NAME	START IP	END IP	...
all	0.0.0.0	255.255.255.255	...

At the bottom, a note states: 'Connections from the VNET/Subnet specified below provides access to all databases in'.

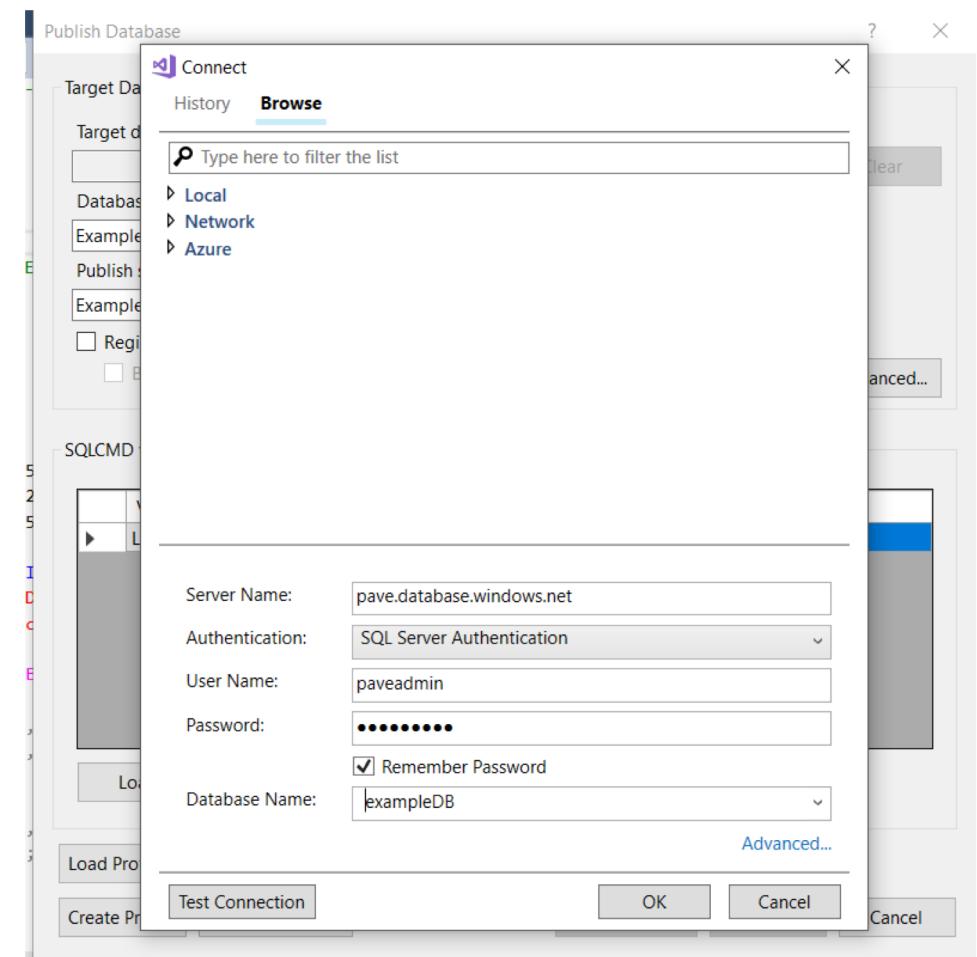
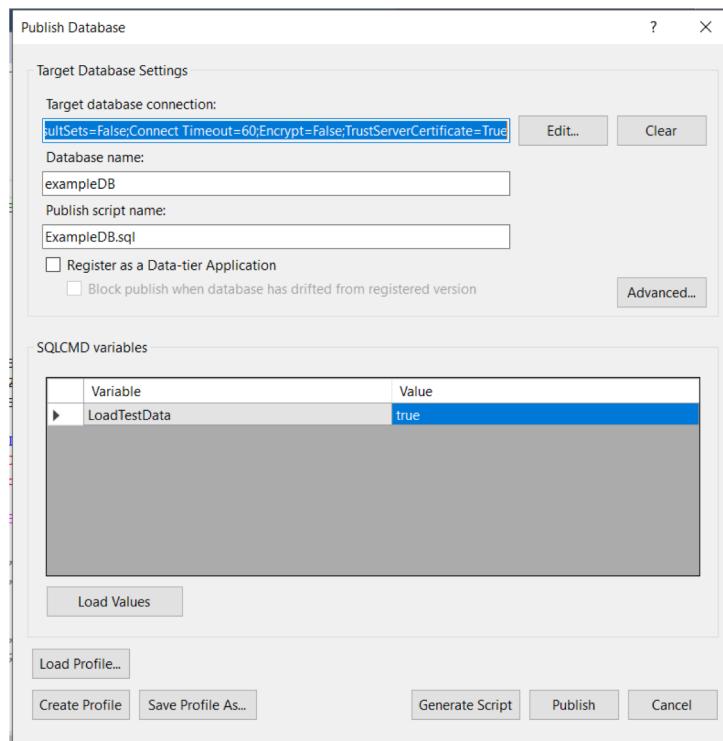
IMPORTANT NOTE

This firewall rule is for ease of development and is **NOT repeat NOT secure** if / when you deploy a database for use in production. You **MUST** get advice from your client / supervisor / employer or someone who understands networking as to the correct firewall rules to put in place.

Networking and Firewall configuration is outside the scope of this course.

Publish From VS To Azure

1. Select Publish Menu Item (must have Built first)
2. (If you want to deploy test data) Set 'LoadTestData' to true
3. Select / Add Target Database Connection
4. Click Publish



Verify From Azure Query Editor

1. Check the expected table shave been created.
2. Check that they are populated with data as expected

The screenshot shows the Azure Query Editor interface. The top bar has a tab labeled "Query 1" with an "X" button. Below the bar are two buttons: "Run" and "Cancel query". The main area contains a code editor with the following SQL query:

```
1 select e.*, s.FirstName, su.Title
2 from enrollment e
3 inner join student s
4 on e.studentid = s.studentid
5 inner join subject su
6 on e.code = su.code; |
```

To the right of the code editor is a results pane. It has tabs for "Results" and "Messages", with "Results" selected. A search bar at the top of the results pane says "Search to filter items...". The results table has the following columns: STUDENTID, CODE, YEAR, SEMESTER, RESULT, FIRSTNAME, and TITLE. The data in the table is:

STUDENTID	CODE	YEAR	SEMESTER	RESULT	FIRSTNAME	TITLE
s12345678	ICTDBS501	1997	1	SEN	Michael	Advanced Database
s12345678	ICTPRG501	1996	2	D	Michael	Advanced OO Programming
s23456789	ICTDBS501	1994	1	SEN	Kurt	Advanced Database
s34567890	ICTDBS501	2011	1	SEN	Amy	Advanced Database

At the bottom of the results pane, there is a yellow bar with a green checkmark icon and the text "Query succeeded | 4s".

Productivity – What we expect of you.

After a couple of practice runs you **should** be able to:

Build & Deploy a (simple to moderately complex) Database inside of 1 – 2 hours.

When combined with Entity Framework (covered in seminar) you **should** be able to:

Build & Deploy a (simple to moderately complex) Database

AND

Build & Deploy a RESTful Web API interface to that database

Inside a total 3 - 4 hours