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

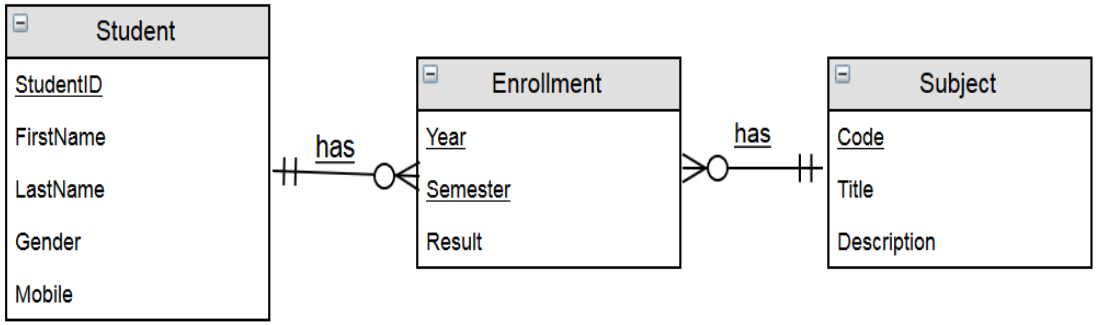
Now it is time to get more ‘practical’

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

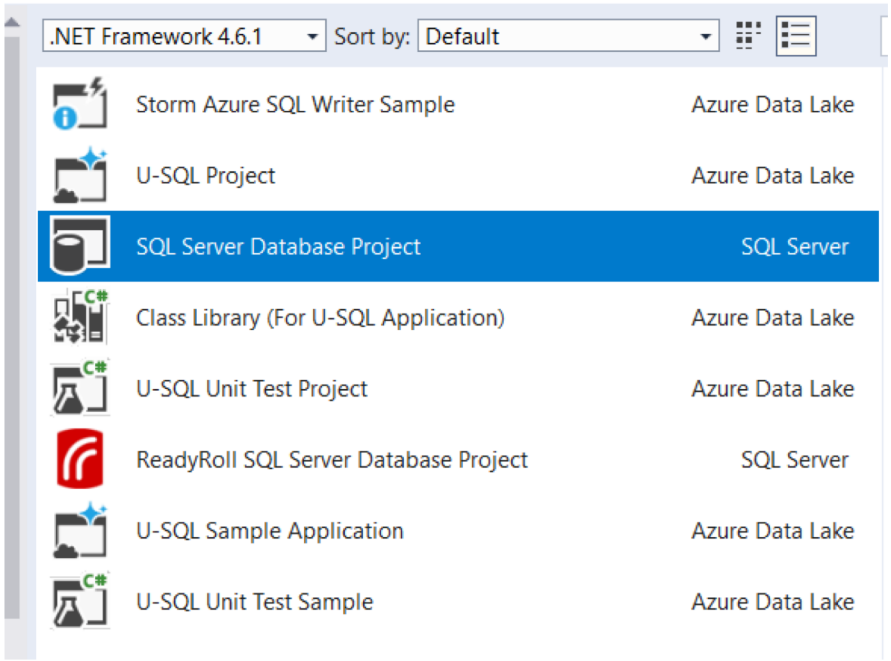
STUDENT (STUDENTID, FIRSTNAME, LASTNAME, GENDER, MOBILE)

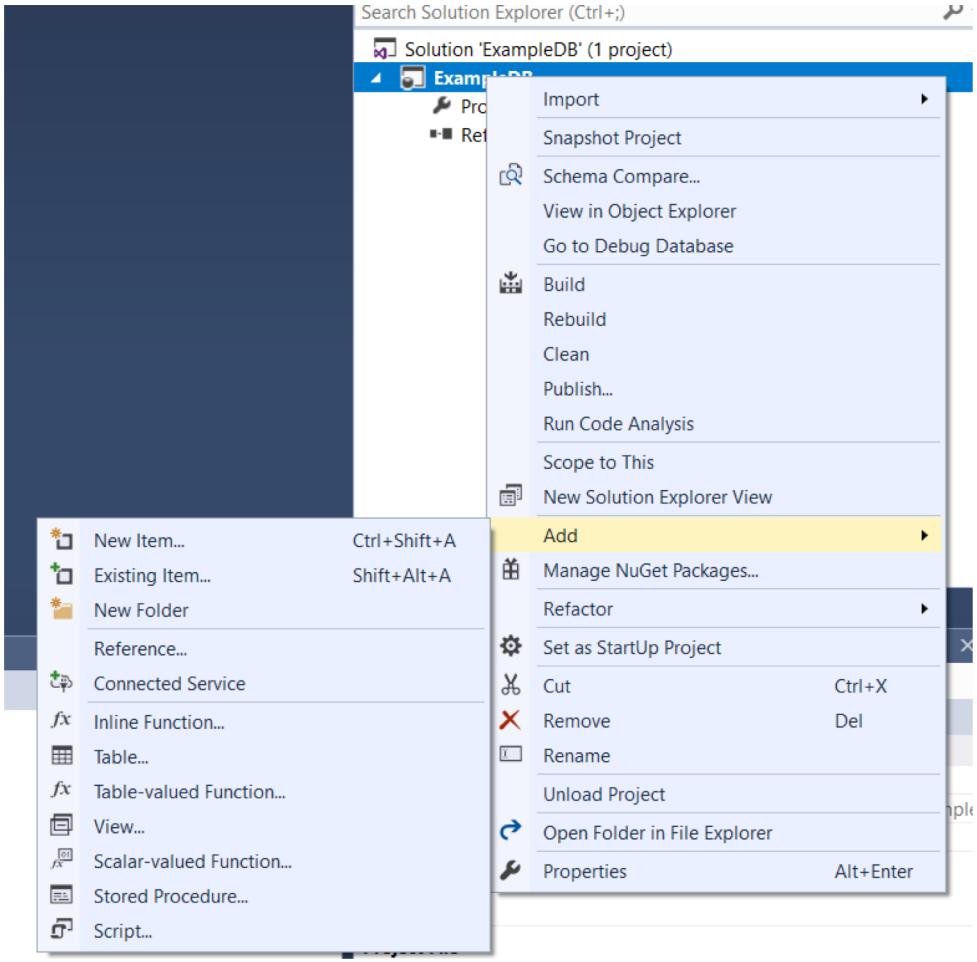
SUBJECT (CODE, TITLE, DESCRIPTION)

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

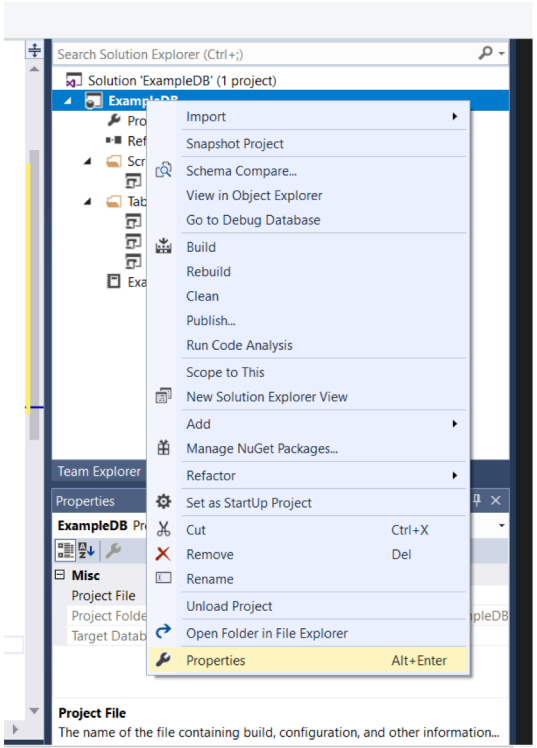
2. Create a DB Project in VS & Add Tables

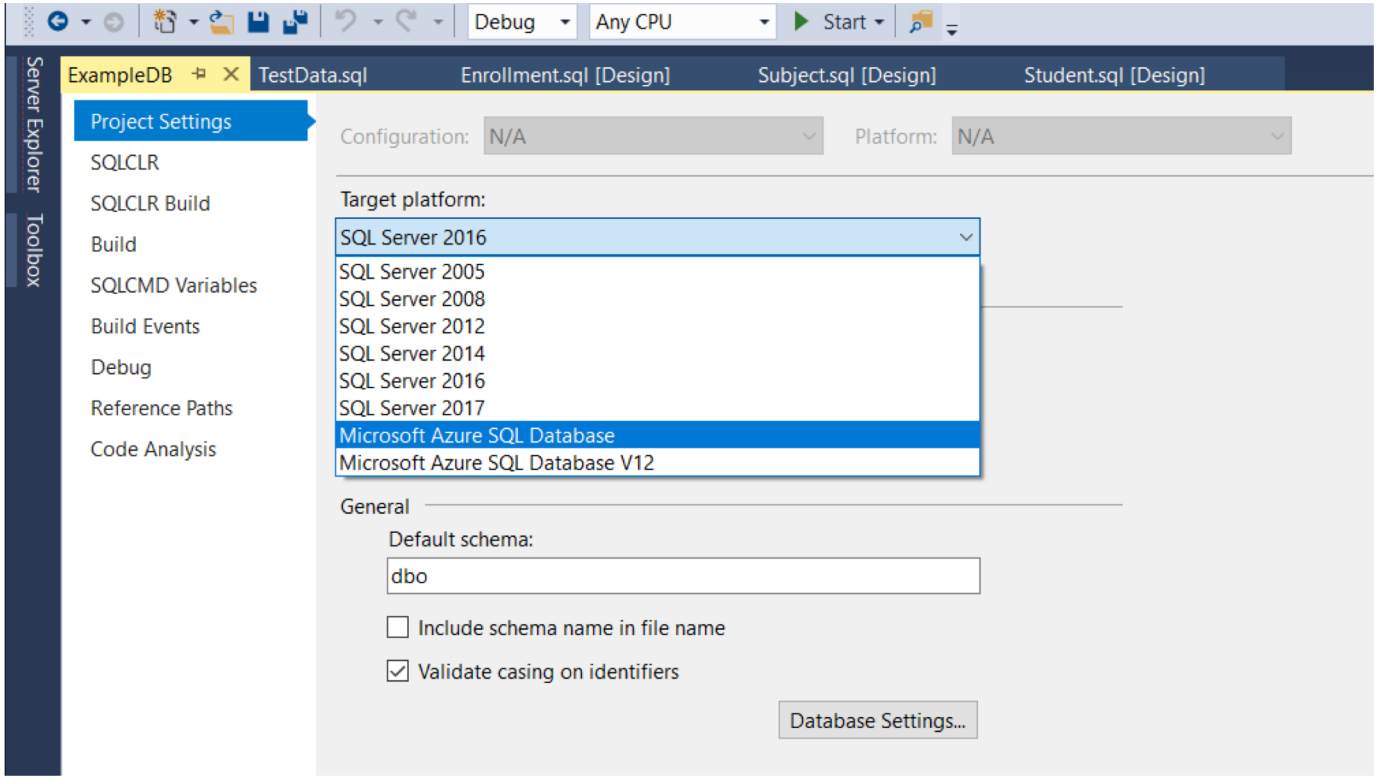




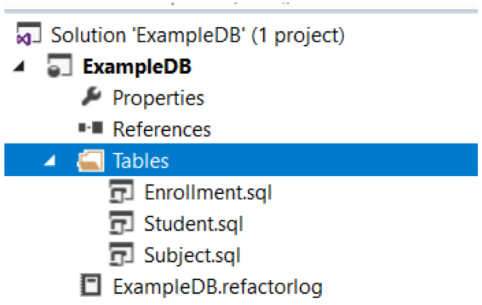
Make it an Azure Database

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





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

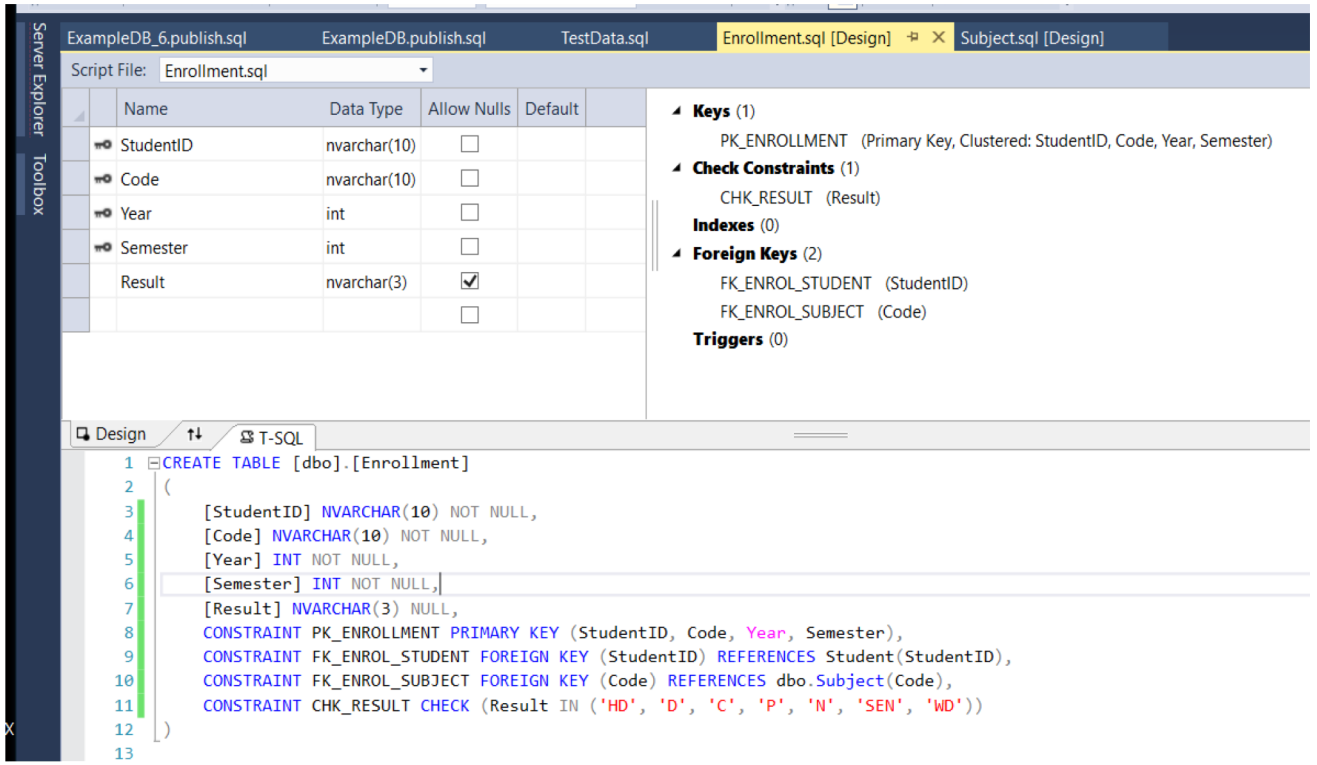


Design | T-SQL

```
CREATE TABLE [dbo].[Student]
(
    [StudentID] NVARCHAR(10) NOT NULL,
    [FirstName] NVARCHAR(50) NOT NULL,
    [LastName] NVARCHAR(50) NOT NULL,
    [Gender] NVARCHAR(1) NULL,
    [Mobile] INT NULL,
    CONSTRAINT PK_STUDENT PRIMARY KEY (StudentID),
    CONSTRAINT CHK_GENDER CHECK (Gender IN ('M', 'F'))
)
```

Design | T-SQL

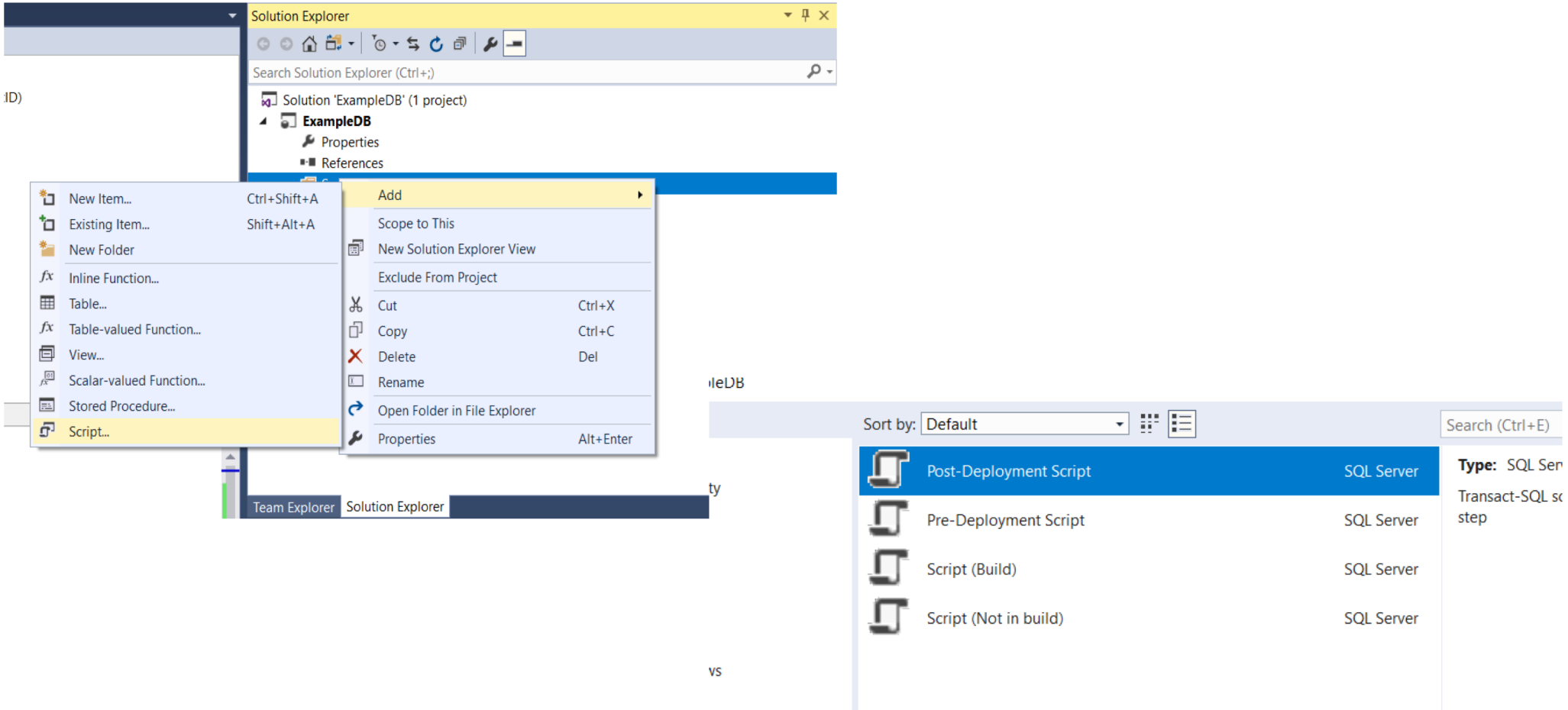
```
CREATE TABLE [dbo].[Subject]
(
    [Code] NVARCHAR(10) NOT NULL,
    [Title] NVARCHAR(50) NOT NULL,
    [Description] NVARCHAR(200) NULL,
    CONSTRAINT PK_SUBJECT PRIMARY KEY (Code)
)
```



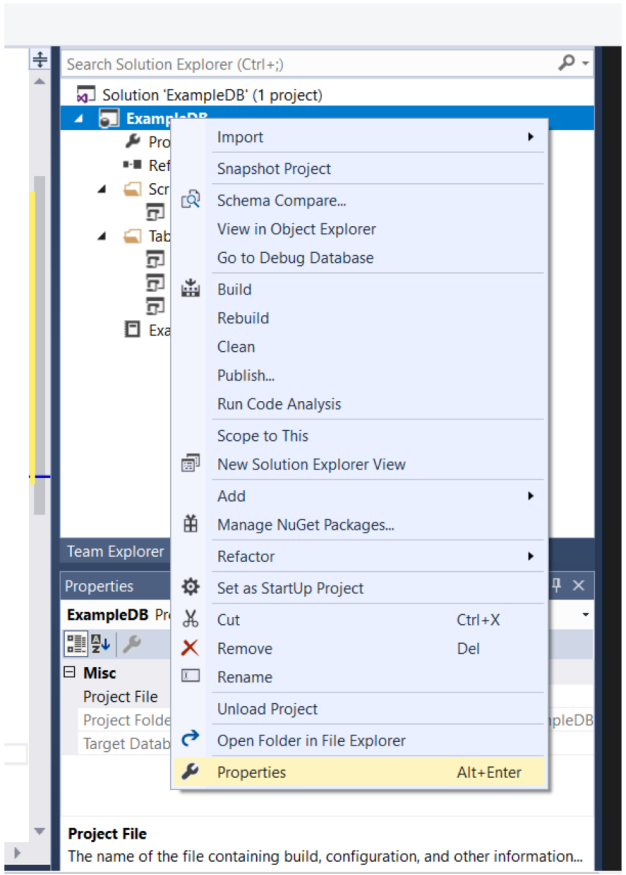
Design | T-SQL

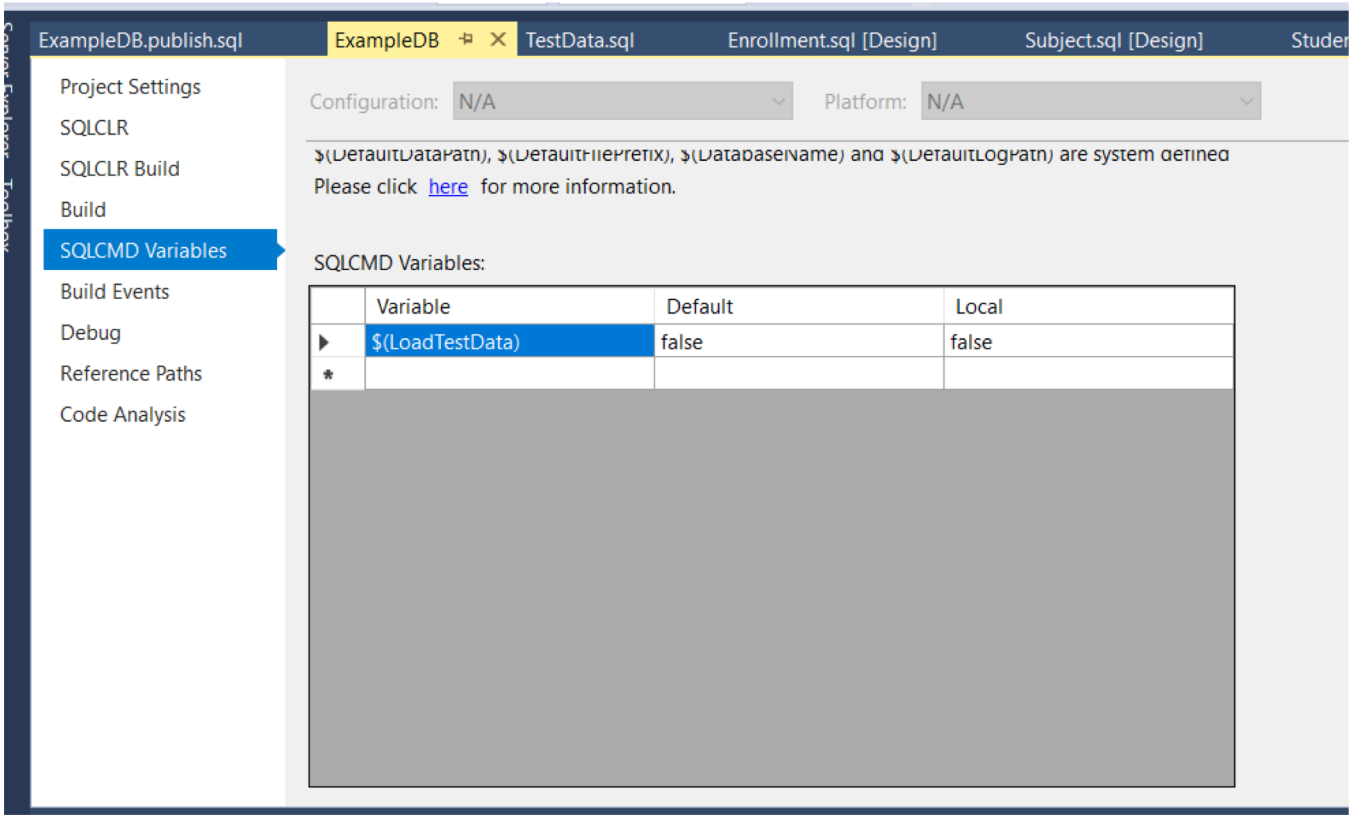
```
CREATE TABLE [dbo].[Enrollment]
(
    [StudentID] NVARCHAR(10) NOT NULL,
    [Code] NVARCHAR(10) NOT NULL,
    [Year] INT NOT NULL,
    [Semester] INT NOT NULL,
    [Result] NVARCHAR(3) NULL,
    CONSTRAINT PK_ENROLLMENT PRIMARY KEY (StudentID, Code, Year, Semester),
    CONSTRAINT FK_ENROL_STUDENT FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
    CONSTRAINT FK_ENROL_SUBJECT FOREIGN KEY (Code) REFERENCES dbo.Subject(Code),
    CONSTRAINT CHK_RESULT CHECK (Result IN ('HD', 'D', 'C', 'P', 'N', 'SEN', 'WD'))
)
```

Create a Post Deployment (PD) Script



Set SQLCMD Variable for Loading Test Data





Add Test Data in Post Deployment Script

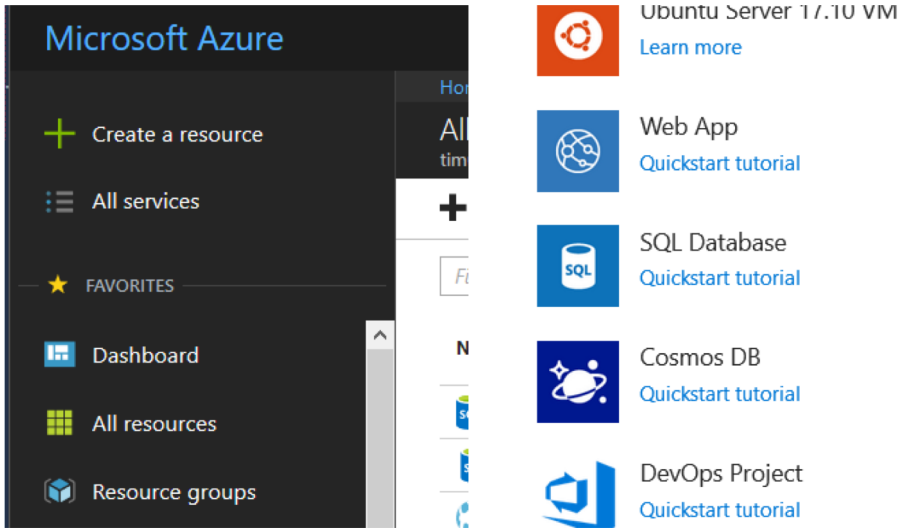
Note the conditional logic around the SQLCMD variable $(LoadTestData)

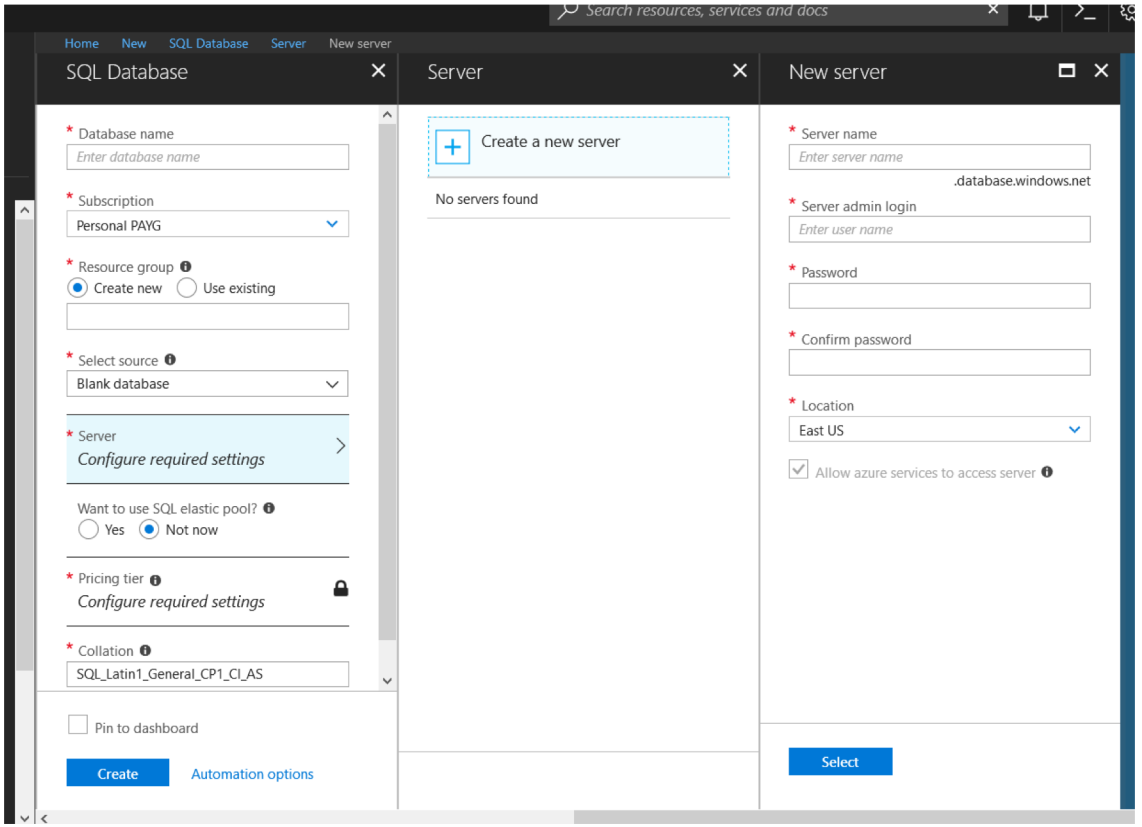
Also note the BEGIN and END tags

```
11  */
12
13  IF '$(LoadTestData)' = 'true'
14
15  BEGIN
16
17  -- ENSURE THERE IS NO DATA IN THE TABLES BEFORE DEPLOYTING TEST DATA
18  DELETE FROM ENROLLMENT;
19  DELETE FROM STUDENT;
20  DELETE FROM SUBJECT;
21
22  -- INSERT THE TEST DATA
23  INSERT INTO STUDENT (STUDENTID, FIRSTNAME, LASTNAME, GENDER, MOBILE) VALUES
24  ('S12345678', 'Michael', 'Hutchence', 'M', 0400555123),
25  ('s23456789', 'Kurt' , 'Cobain', 'M', 0411555321),
26  ('s34567890', 'Amy', 'Winehouse', 'F', 0422555432),
27  ('s98765432', 'Janis', 'Joplin', 'F', 0433555543);
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  INSERT INTO ENROLLMENT (STUDENTID, CODE, YEAR, SEMESTER, RESULT) VALUES
34  ('s12345678', 'ICTPRG501', 1996, 2, 'D'),
35  ('s12345678', 'ICTDBS501', 1997, 1, 'SEN'),
36  ('s23456789', 'ICTDBS501', 1994, 1, 'SEN'),
37  ('s98765432', 'ICTPRG501', 1969, 2, 'N'),
38  ('s98765432', 'ICTPRG501', 1970, 1, 'C'),
39  ('s98765432', 'ICTDBS501', 1970, 2, 'SEN'),
40  ('s34567890', 'ICTDBS501', 2011, 1, 'SEN');
41
42  END;
```

Create DB on Azure

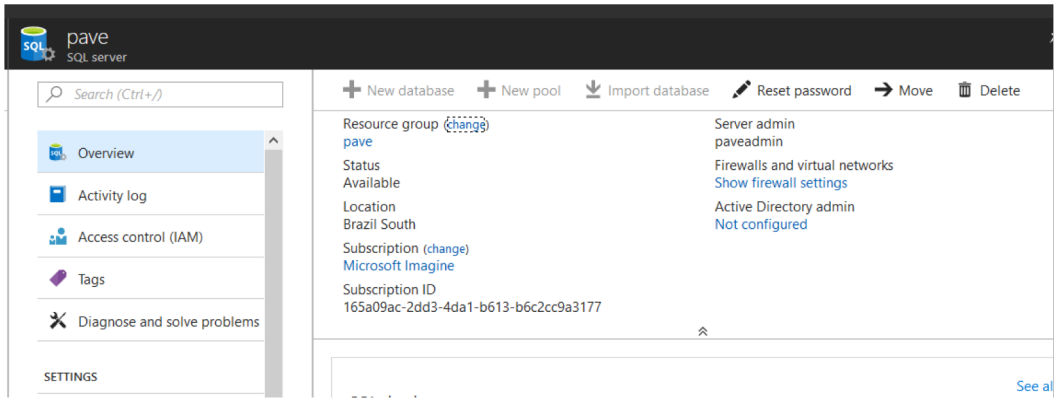
1. Select Create a Resource
2. Select SQL Database
3. Enter the database, & server details
4. Wait for it to deploy



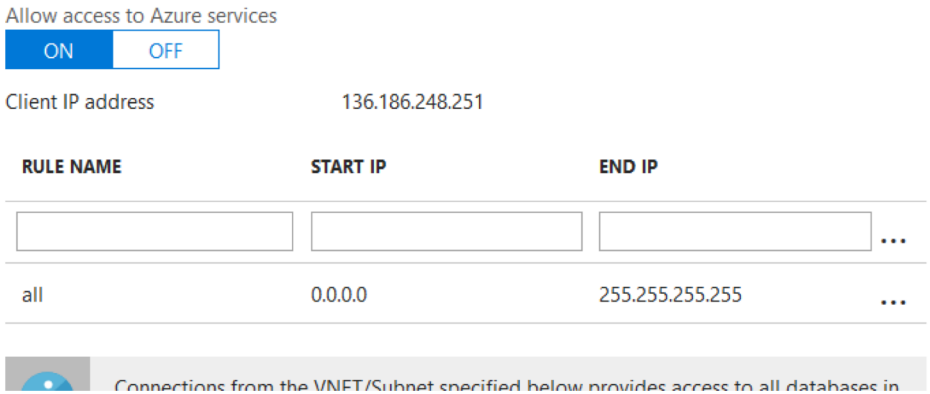


Azure Firewall Rules

On the server settings page select Show Firewall Settings



Add a firewall rule as shown below from 0.0.0.0 to 255.255.255.255



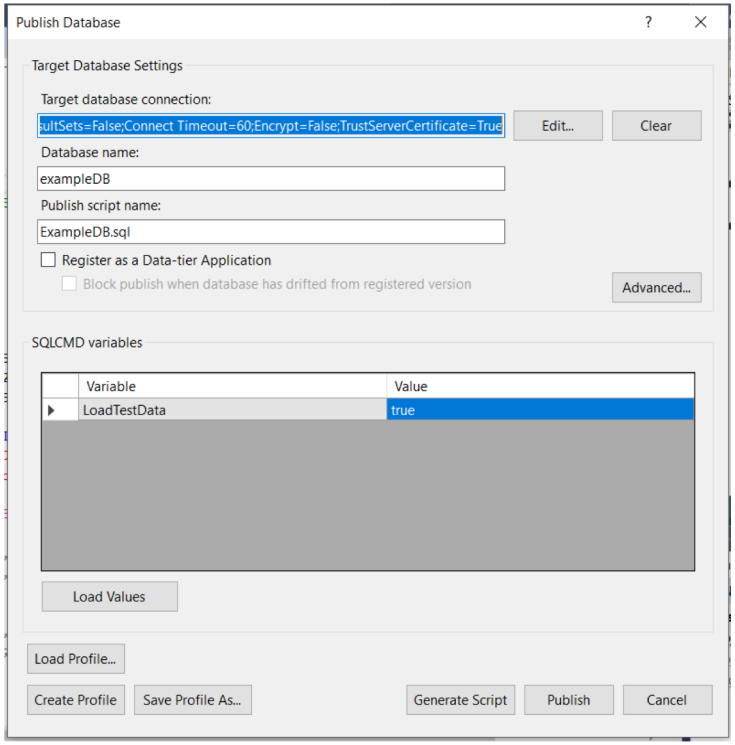
IMPORTANT NOTE

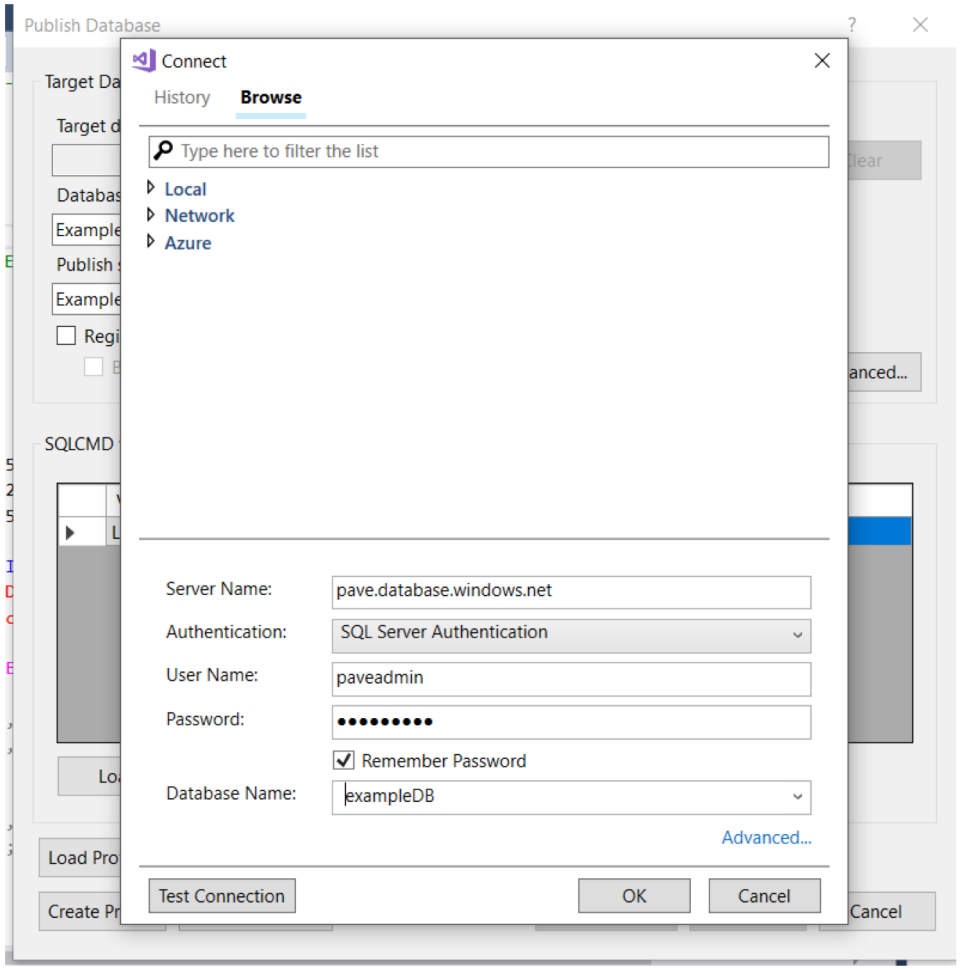
This firewall rule is for ease of development and is NOT repeat NOT secure if / when you deploy a database for use in production
You MUST get advice from your client / supervisor / employer or someone who understands networking as to the correct firewall rules to put in place.

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

Query 1

Run Cancel 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;
```

Results Messages

Search to filter items...

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

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