Querying with Transact-SQL

Getting Started with Azure SQL Database / SQL Server

Overview

Transact-SQL is an essential skill for database professionals, developers, and data analysts working with Microsoft SQL Server or Microsoft Azure SQL Database. This course combines online presentations with hands-on labs that will give you practical experience and a chance to test and extend your Transact-SQL programming skills.

The labs in this course and based on the **AdventureWorksLT** sample database. A hosted lab environment is provided for the lab exercises, so you don't need to set up this database to complete the labs; but if you want to explore the database and experiment with the Transact-SQL demonstrations that are shown in the lectures, you'll need a database of your own. This document explains how to achieve this using Microsoft Azure SQL Database, a cloud-based relational database service. This is the recommended environment for getting started with Transact-SQL as it requires minimal software installation and configuration on your computer. However, if you prefer, you can install a local instance of SQL Server Express and download and attach the sample database – there are instructions at the end of this document for doing this.

What You'll Need

- A web browser
- A Microsoft account
- A Microsoft Azure subscription
- Optionally, a Windows, Mac OS X, or Linux computer with a SQL Server client tool installed.

Note: Alternatively, you can install SQL Server Express and SQL Server Management Studio on a Windows client computer and host the database locally – there are hiogh-level instructions for this at the end of this document. However, installing and configuring a database management system like SQL Server is a complex task – unless you're confident in your ability to troubleshoot installation issues on your specific computer environment, we recommend using Azure SQL Database.

Creating an Azure Subscription

If you already have a Microsoft Azure subscription, you can skip this section.

If you have never had an Azure subscription, you can create a 30-day free trial subscription, which includes $200 of free credit in your local currency, which is comfortably enough to complete the labs in this course. Alternatively, you may be able to join the Visual Studio Dev Essential program, which includes a 12-month Azure subscription with $25 per month of credit included. You will need to provide a valid credit card number for verification, but you will not be charged for Azure services – for more information, see the frequently asked questions in the Azure sign-up page.

If you have previously had a free trial subscription that has now expired, you will not be able to sign up for a second free trial. In this case, you must either use a Visual Studio Dev Essentials Azure account, or sign up for a pay-as-you-go subscription. The lab instructions in this course are designed to minimize the cost of the Azure resources required to complete the hands-on exercises.

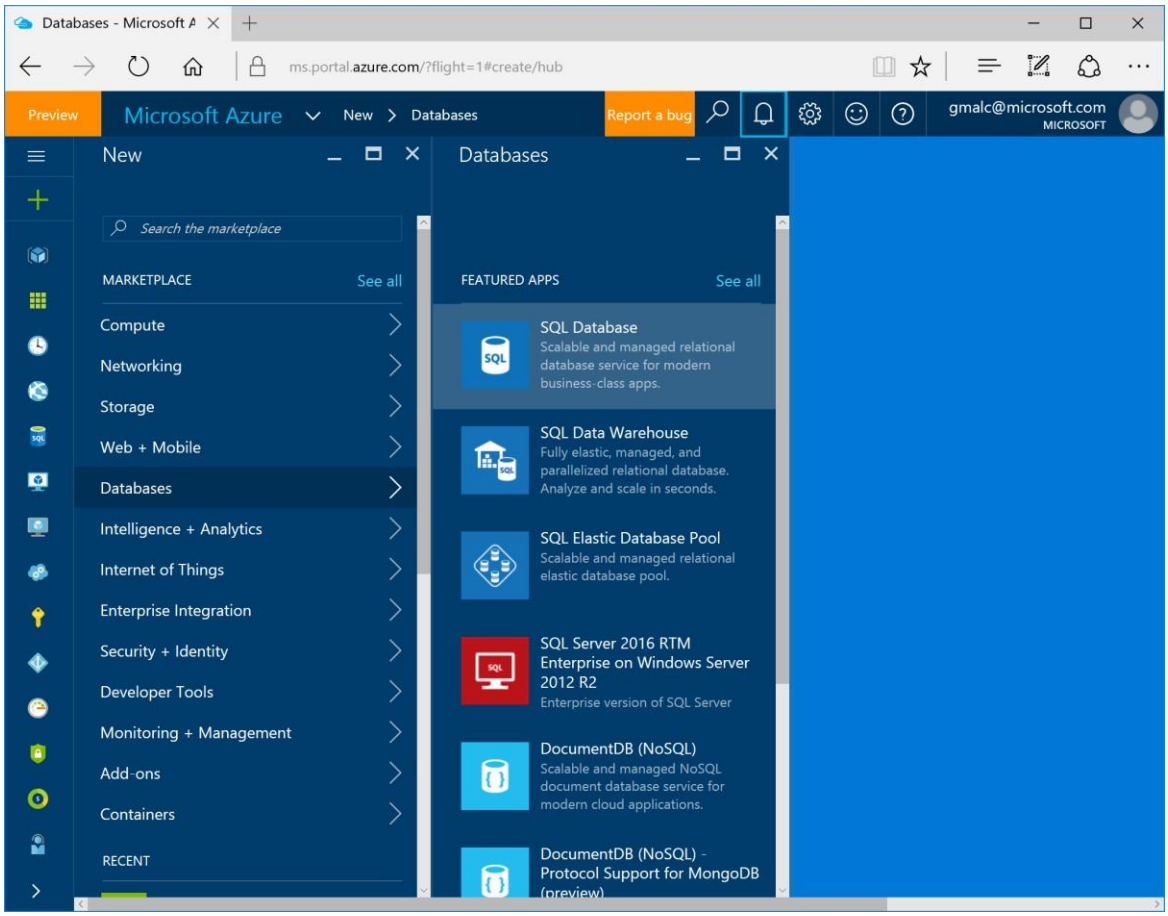
To set up an Azure subscription:

1. If you already have a Microsoft account that has not already been used to sign up for a free Azure trial subscription, you're ready to get started. If not, don't worry, just create a new Microsoft account at https://signup.live.com.
2. After you've created a Microsoft account, browse to http://aka.ms/edx-dat201x-az and start a free trial. Then follow the instructions to sign up for a free 30-day trial subscription to Microsoft Azure. Alternatively, after creating your Microsoft account, go to https://azure.microsoft.com/en-us/pricing/member-offers/vs-dev-essentials/ and sign up for the Visual Studio Dev Essentials program, and then activate your Azure subscription. You'll need to sign-in with your Microsoft account if you're not already signed in. When activating either a 30-day trial or a Visual Studio Dev Essentials Azure subscription, you will need to provide valid payment details – don't worry, your credit card won't be charged for any services you use during the trial period, and the account is automatically deactivated at the end of the trial period unless you explicitly decide to keep it active.

Create an Azure SQL Database

Now that you have an Azure subscription, you can create an Azure SQL Database instance to use in the labs.

1. Browse to http://portal.azure.com. If you are prompted to sign in, do so with the Microsoft account that is associated with your Azure subscription.
2. At the bottom of the Hub menu (the vertical bar on the left), click **New** (represented by a **+** symbol if the menu is minimized), and then in the **New** blade that appears, click **Databases**, and then click **SQL Database**.

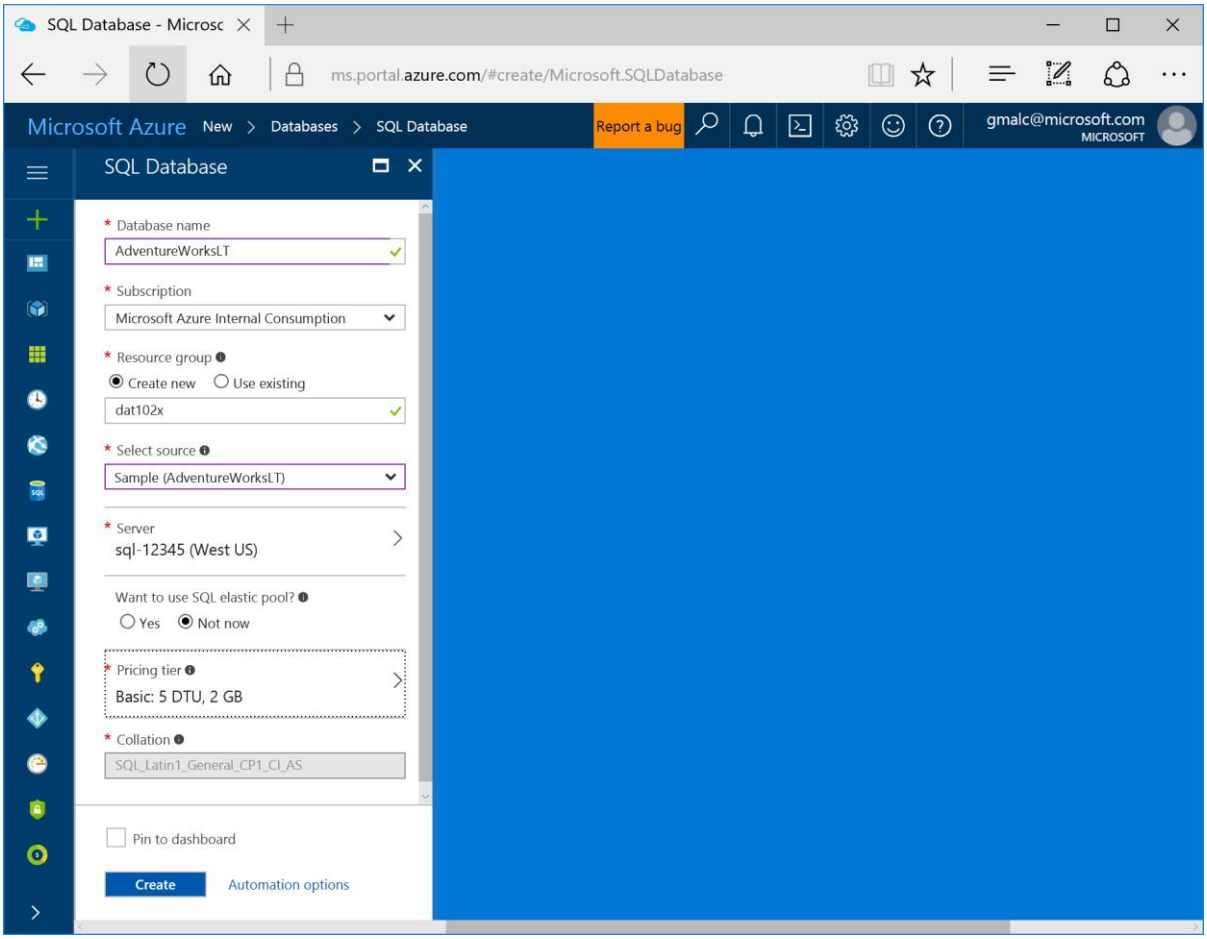


3. In the **SQL Database** blade:
    a Enter the name **AdventureWorksLT**
    b In the **Subscription** box, ensure that your subscription is listed.
    c In the **Resource group** section, ensure that **New** is selected, and enter any unique name as the new resource group name.
    d In the **Select Source** list, select **Sample (AdventureWorksLT)**.
    e Click **Server**. Then click **Create a new server** and enter the following details and click **OK**.
        - A unique Server name for your server (a pink exclamation mark will be displayed if the name you have entered is invalid or already in use, otherwise a green tick is shown)
        - A user name you want to assign to the Server admin login. This can be your name or some other name you'll remember easily – however, you cannot use "Administrator".
        - A password for your server administrator account. This must meet the password complexity rules for Azure SQL database, so for example it cannot be blank or "password". You must confirm the password in a second text box.
        - The location where your server should be hosted. Choose the location nearest to you.

- Leave the option to allow **Azure services to access the server** selected (this opens an internal firewall port in the Azure datacenter to allow other Azure services to use the database).

f In the **Pricing Tier** section, select **Basic**.

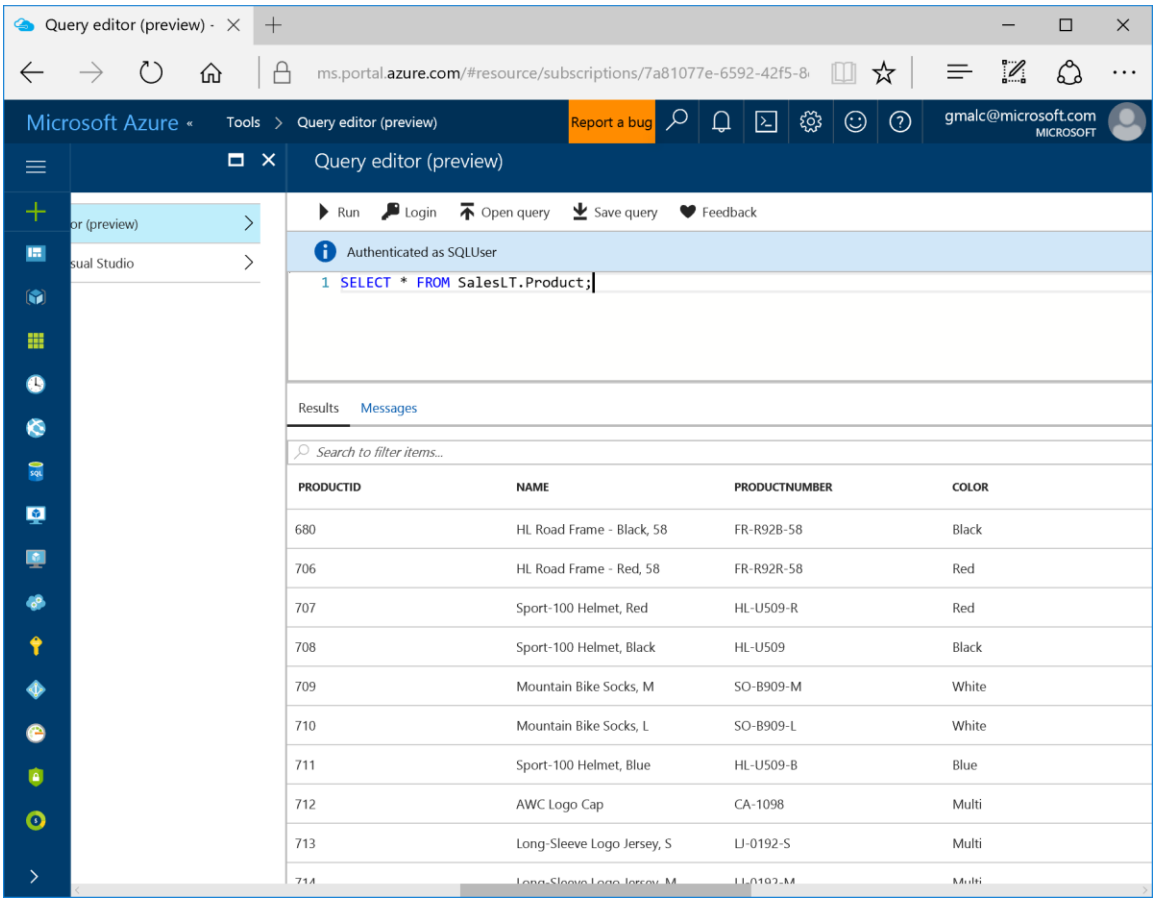
g Ensure that your selections are similar to those below, and click **Create**.



4. After a short time, your SQL Database will be created, and a notification is displayed on the dashboard. To view the blade for the database, click **All Resources**, and then click the **AdventureWorksLT** database.
5. On the **AdventureWorksLT** blade, click **Tools**. Then on the **Tools** blade, click **</> Query editor**. This opens the web-based query interface for your Azure SQL Database.
6. In the toolbar for the query editor, click **Login**, and then log into your database using SQL Server authentication and entering the user name and password you specified when provisioning the Azure SQL Database server.
7. In the query editor, enter the following Transact-SQL query to retrieve the contents of the **SalesLT.Product** table in the **AdventureWorksLT** database:

```
SELECT * FROM SalesLT.Product;
```

8. Click **Run**, and review the results (which show the product records in the table), as shown below.



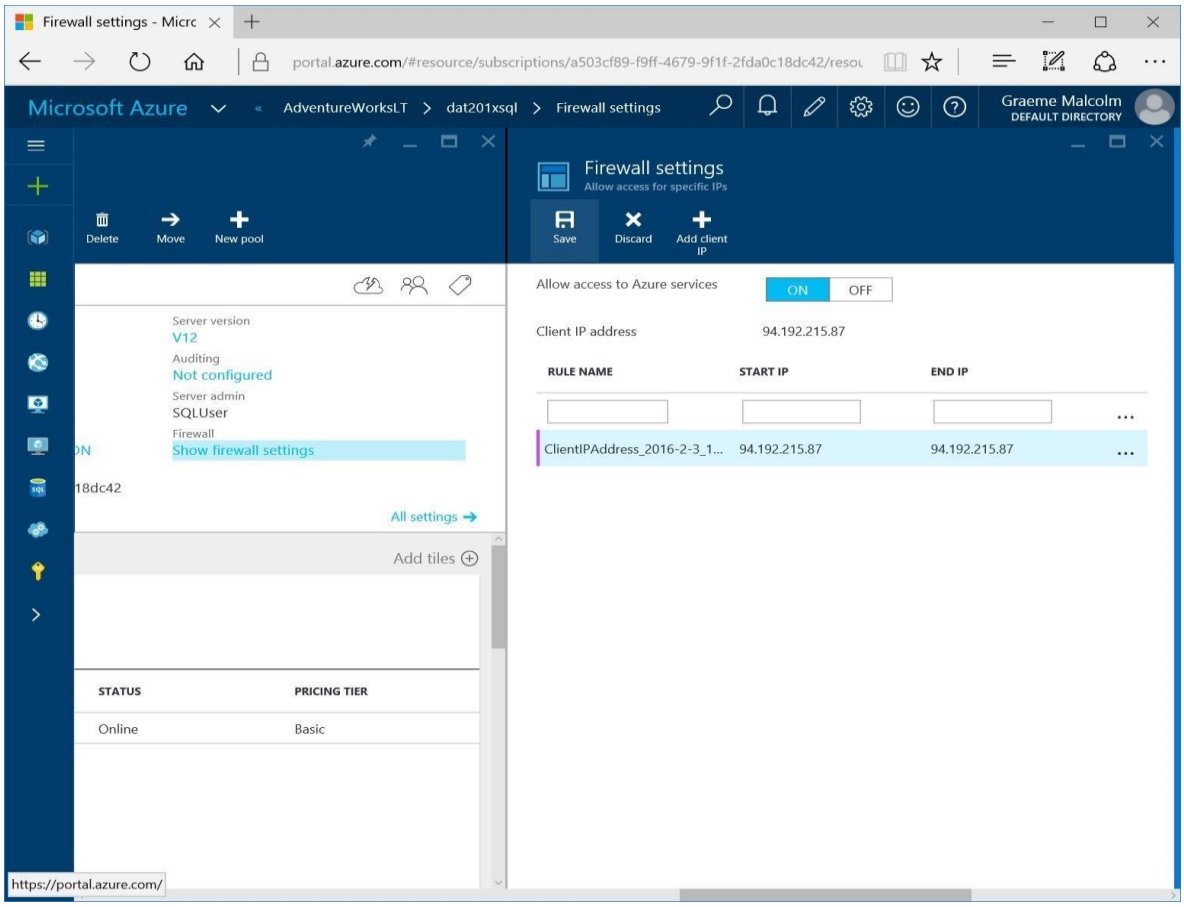
Optional – Install a Client Tool

You will be able to use the browser-based query editor to run Transact-SQL queries in the database and view the results. However, for a richer experience, you may want to install a client tool on your own computer and use it to connect to your Azure SQL Database.

Configure Firewall Rules for your Azure SQL Database Server

Azure SQL Databases are protected by a firewall, so you must configure a rule that permits access from your client computer.

1. In the **AdventureWorksLT** blade, under **Essentials**, click the server name for your database server (which should be in the format ***server_name*.database.windows.net**).
2. In the blade for your SQL server, under **Essentials**, click **Show firewall settings**.
3. In the **Firewall settings** blade, click the **Add client IP** icon to create a firewall rule for your client computer, and then click **Save**.



Note: If your computer's public-facing IP address changes (or you want to use a different computer), you'll need to repeat this step to allow access. Alternatively, you can modify the firewall settings for your Azure SQL Database server to allow a range of IP addresses – see the Azure SQL Database documentation for details of how to do this.

Installing and Connecting from a Client Tool

You can use either of the following tools to develop your Transact-SQL queries.

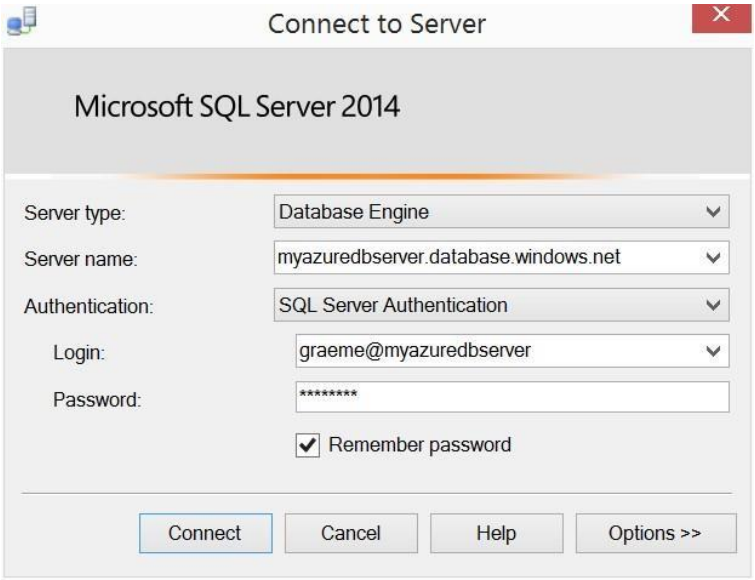
Microsoft SQL Server Management Studio

SQL Server Management Studio is the primary management tool for Microsoft SQL Server from Windows-based client computers, and you can also use it to manage and query Azure SQL Database.

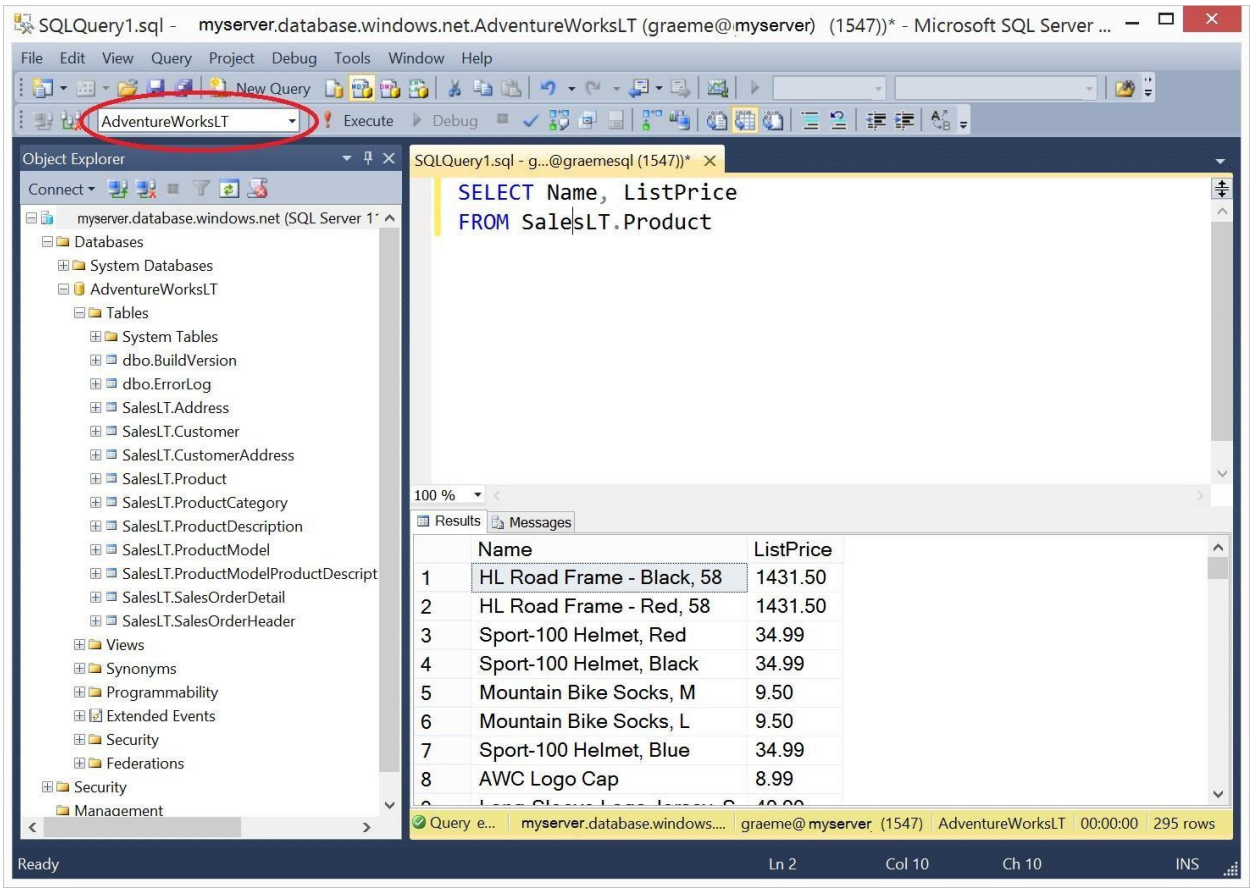
If you do not already have SQL Server Management Studio installed, you can download it from https://aka.ms/edx-dat201x-ssms.

When the download is complete, run the executable file to install SQL Server management Studio.

After installing SQL Server Management Studio, you can start it and connect to your Azure SQL Database server by selecting the option to use SQL Server authentication, specifying the fully-qualified name of your Azure SQL Database server (**<*your_server_name*>.database.windows.net**), and entering your user name in the format **<*your_user_name*>@<*your_server_name*>** and password, as shown here:



After connecting, you can create a new query and run it by clicking **Execute**, and you can save and open Transact-SQL scripts. Be sure to select the **AdventureWorksLT** database when running your queries as shown here:

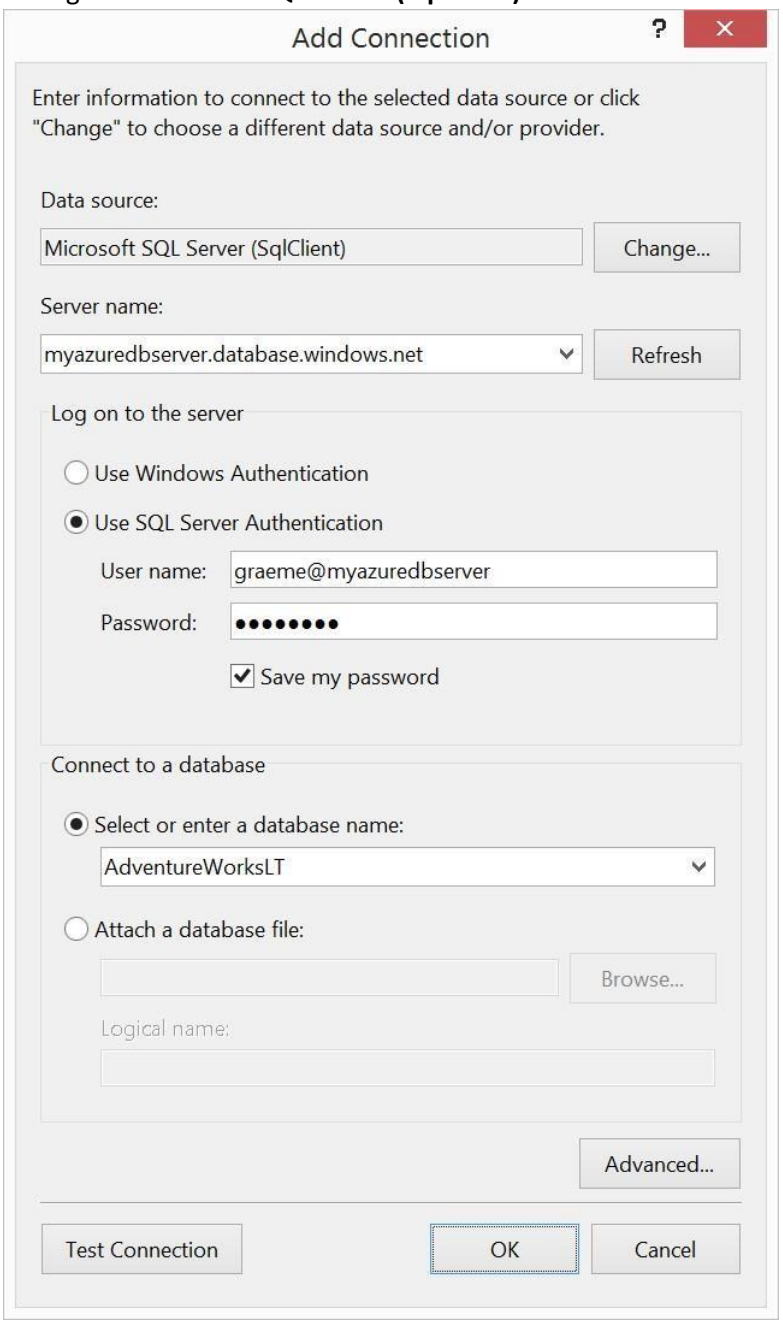


Microsoft Visual Studio

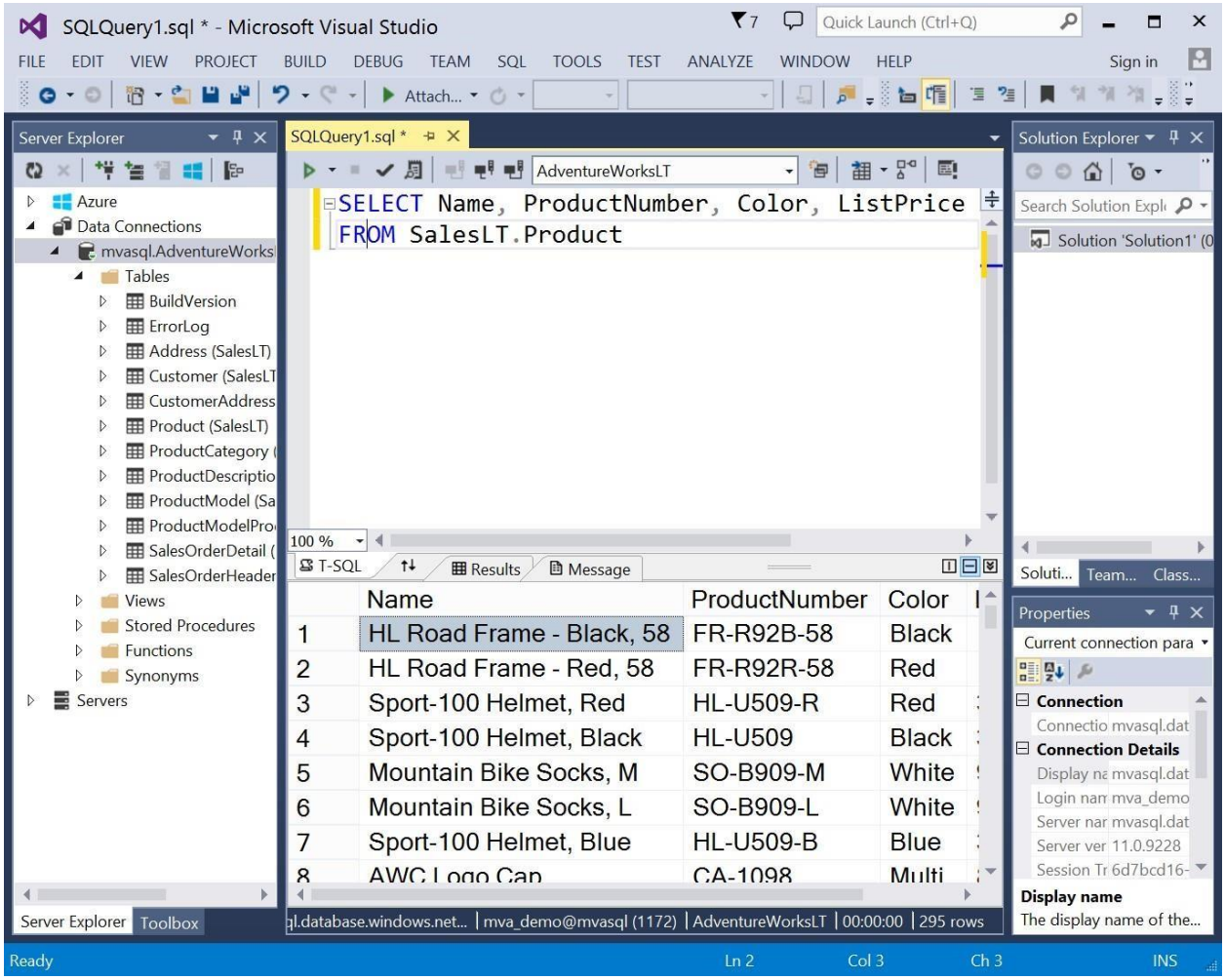
If you are primarily a developer, you may prefer to use Visual Studio to create your Transact-SQL queries. Visual Studio is a comprehensive software development environment for all kinds of software

projects, including database development. You can download the free Community edition of Visual Studio from http://aka.ms/edx-dat201x-vs and install it on your Windows computer.

When you install Visual Studio, be sure to select the option to include the **SQL Server Data Tools** optional component. Then, in the Server Explorer pane, you can create a data connection to your Azure SQL database server using the **Microsoft SQL Server (SqlClient)** data source as shown here.



After you have created a data connection, you can view database objects in the Server Explorer window. You can also create and save Transact-SQL scripts and run queries, as shown here.



Alternative Client Tools for Non-Windows Computers

If you are using a non-Windows computer, you will be unable to install SQL Server Management Studio or Visual Studio. However, you can use the following options to perform the labs. Microsoft provides no endorsement or support for non-Microsoft client tools, and you install and use them at your own discretion.

If you are using a Mac OS X computer, you can:

- Use virtualization software such as Parallels to install a virtualized instance of Windows on your Mac, then install SQL Server Management Studio or Visual Studio and connect to Azure SQL Database.
- Install a third-party SQL Server client GUI tool of your choice, or the command line cross-platform sqlcli tool* (https://www.npmjs.com/package/sql-cli) and connect to Azure SQL Database.

If you are using Linux, you can install a third-party SQL Server client GUI tool of your choice or the command line cross-platform sql-cli tool* (https://www.npmjs.com/package/sql-cli) and connect to Azure SQL Database.

*Note that the cross-platform sql-cli tool is a command-line interface, and does not support many of the features in graphical tools such as SQL Server Management Studio. In particular, you may need to enter Transact-SQL statements on a single line. To view help for the sql-cli tool, enter the command **mssql -h**.

Troubleshooting Connection Errors

After you have installed a client tool, you can connect to Azure SQL Database across the Internet. To establish a connection, you must use the following connection information:

- **Server**: <*server_name*>.database.windows.net
- **Authentication Mode**: SQL Server authentication (sometimes called *Native* authentication)
- **Login**: <*user_name*>@<*server_name*> (you can omit @<*server_name*> in some Windows based clients)
- **Port**: 1433 (this is the default in most SQL Server client tools) Most connection errors are caused by:
- Incorrect server name or login credentials – check for typing mistakes and capitalization.
- Firewall restrictions – ensure that you have created a firewall rule for your Azure SQL Database server that permits access from your local computer. If this doesn't work, try creating a firewall rule for a range of IP addresses – as a last resort, try 0.0.0.0 to 255.255.255.255 (which allows access from any Internet-connected computer – this is not recommended for a real production server, but should be OK if your server only contains the sample database for the labs in this course).

  If configuring the firewall in Azure still doesn't resolve the issue, there may be a firewall on your local network or computer that's preventing the connection. If you're using a school or corporate network, speak to your network administrator. If you have a firewall enabled on your local computer, refer to the documentation provided by the supplier and enable outbound connections to port 1433 from your client tool. Alternatively, you may be able to temporarily disable your firewall to establish the connection, and then re-enable it after you are connected – if you choose to do this, you do so at your own risk.

Alternative Setup using SQL Server Express

For the best experience, sign up for a free Azure trial subscription and follow the instructions provided above. If you are unable to create an Azure subscription, you can use the following instructions to install SQL Server Express on a Windows computer, and deploy a sample database that is similar to the one used in the course demonstration and labs.

Note: Installing a SQL Server database instance can be a complex task, and if you've not worked with SQL Server before, we highly recommend that you use Azure SQL Database instead of installing SQL Server locally.

Install SQL Server Express

SQL Server Express is a free instance of the SQL Server database engine. You can use this to host the sample database used in the labs.

1. Browse to http://aka.ms/edx-dat201x-sql and download and run the installer.
2. Choose a **Custom** installation option, and download the installation media to your local computer.

3. If the **SQL Server Installation Center** window does not appear, in the folder where you extracted the files, run **Setup.exe**. Then, in the **SQL Server Installation Center** window, on the **Installation** page, click **New SQL Server stand-alone installation or add features to an existing installation**.
4. In the SQL Server Setup window; if there are any issues, resolve them by installing any prerequisite software or making any required configuration changes. Then re-run setup.
5. On the **License Terms** page, accept the license terms and click **Next**.
6. On the **Feature Selection** page, under **Instance Features**, select only the **Database Engine Services** feature, and under **Shared Features**, select only the **Client Tools Connectivity** feature. Ensure that the installation location has sufficient disk space. Then click **Next**.
7. On the **Instance Configuration** page, select **Default instance** and click **Next** (note, if you wish, you can install a named instance instead of a default instance – if you do this, when you connect to your SQL Server instance you must specify the name **(local)*instance_name***.)
8. On the **Server Configuration** page, do not change the default selections (unless you are comfortable configuring service accounts). Just click **Next**.
9. On the **Database Engine Configuration** page, select **Mixed Mode (SQL Server authentication and Windows authentication)**, enter a suitable password for the system administrator account (and make a note of it!), and click **Next**.
10. When installation is complete. Click **Close**.
11. Return to the SQL Server Installation Center window, and click the option to install **SQL Server Management Tools**. Then follow the instructions to download and install the client tools.
12. Close the SQL Server Installation center window.
13. Pin the **SQL Server 2016 Management Studio** app to the taskbar – this will make it easier to find when you want to use it.

Install the AdventureWorksLT Sample Database

1. Browse to http://msftdbprodsamples.codeplex.com/releases/view/55330, and click the link to download **AdventureWorksLT2012_Data** (be careful to choose this download and not any of the others!) Save the **AdventureWorksLT2012_Data.mdf** file to the **Data** folder for the SQL Server Express instance you installed (by default, this is C:\Program Files\Microsoft SQL Server\MSSQL*xx*.MSSQLSERVER\MSSQL\DATA). Note, you may be prompted to confirm that you want to grant your user account permission to access this location.
2. Start SQL Server Management Studio, and when prompted, enter or select the following options and click **Connect**:
    - **Server type**: Database Engine
    - **Server name**: (local) (or (local)*instance_name* if you installed a named instance)
    - **Authentication**: SQL Server Authentication
    - **Login**: sa
    - **Password**: *The password you specified during installation*
3. If the Object Explorer pane is not visible, on the **View** menu, click **Object Explorer**. Then in Object Explorer, right-click **Databases** and click **Attach**.
4. In the **Attach Databases** dialog box, under the **Databases to attach** list, click **Add**. Then browse to the folder where you downloaded **AdventureWorksLT2012_Data.mdf**, select it, and click **OK**.

5. In the **Attach Databases** dialog box, in the **"AdventureWorksLT2012" database details** area, select **AdventureWorksLT2012_log.ldf** and click **Remove**. Then click **OK**.
6. In Object Explorer, expand the databases folder and verify that the **AdventureWorksLT2012** database is listed.
7. On the toolbar, click **New Query**. Then in the **Available Databases** list, ensure that **AdventureWorksLT2012** is select and type the following query: SELECT * FROM SalesLT.Product;
8. On the toolbar, click **Execute**, and verify that a table of product data is returned.
9. Close SQL Server Management Studio without saving any files.