

# Exercise: Use SQL to query Azure SQL Database

6 minutes

Sandbox activated! Time remaining: **3 hr 42 min**

You have used 1 of 10 sandboxes for today. More sandboxes will be available tomorrow.

Contoso has provisioned the SQL database and has imported all the inventory data into the data store. As lead developer, you've been asked to run some queries over the data.


In this exercise, you'll query the database to find how many products are in the database, and the number of items in stock for a particular product.

## Setup

To save time, the database is provisioned and populated running a script. You'll download the script from a GitHub repository. The script performs the following operations:

- Creates an Azure SQL Database server.
- Creates an Azure SQL database attached to the server.
- Opens the firewall to allow SQL traffic from the internet.
- Connects to the database and run a SQL script to create a table and insert data.

1. Run the following *git clone* command in the Cloud Shell to clone the repository that contains the data and setup script in GitHub. The repository is copied to a local folder named *dp-900/sql*.

Bash	 Copy
<pre>git clone https://github.com/MicrosoftLearning/DP-900T00A-Azure-Data-Fundamentals dp-900</pre>	

2. Run the following command to move to the *dp-900/sql* folder.

Bash	 Copy
------	--

```
cd dp-900/sql
```

3. Run the **setup.sh** to create the Azure SQL database and server, as follows:

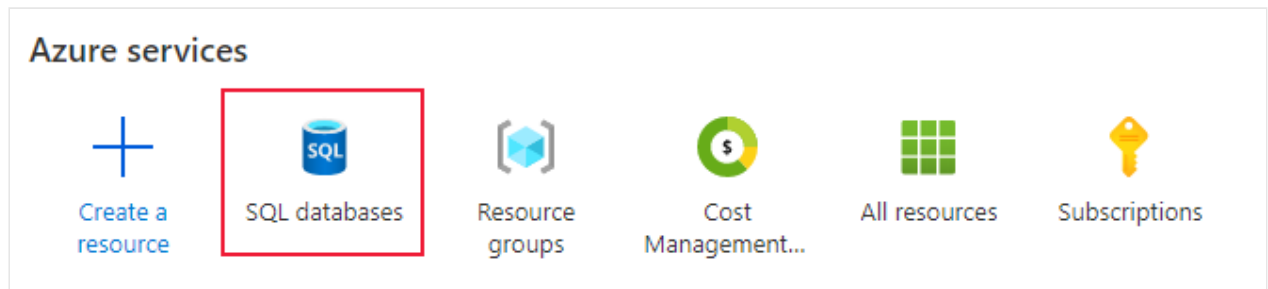
Bash	Copy
bash setup.sh	

The script takes a few minutes to run. When the script finishes, it will display the connection details for the database. Write down the username and password.

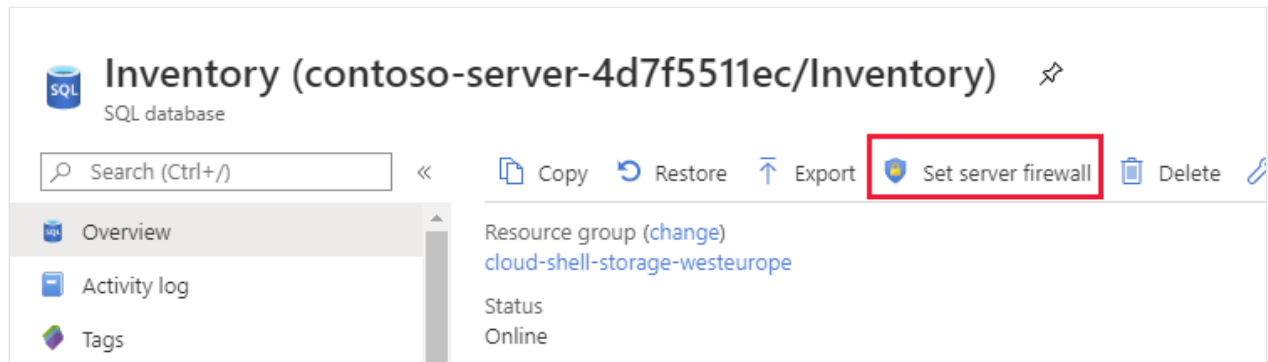
## Connect to the query editor

You'll use the built-in Query editor in the Azure portal to connect to the database and query the data.

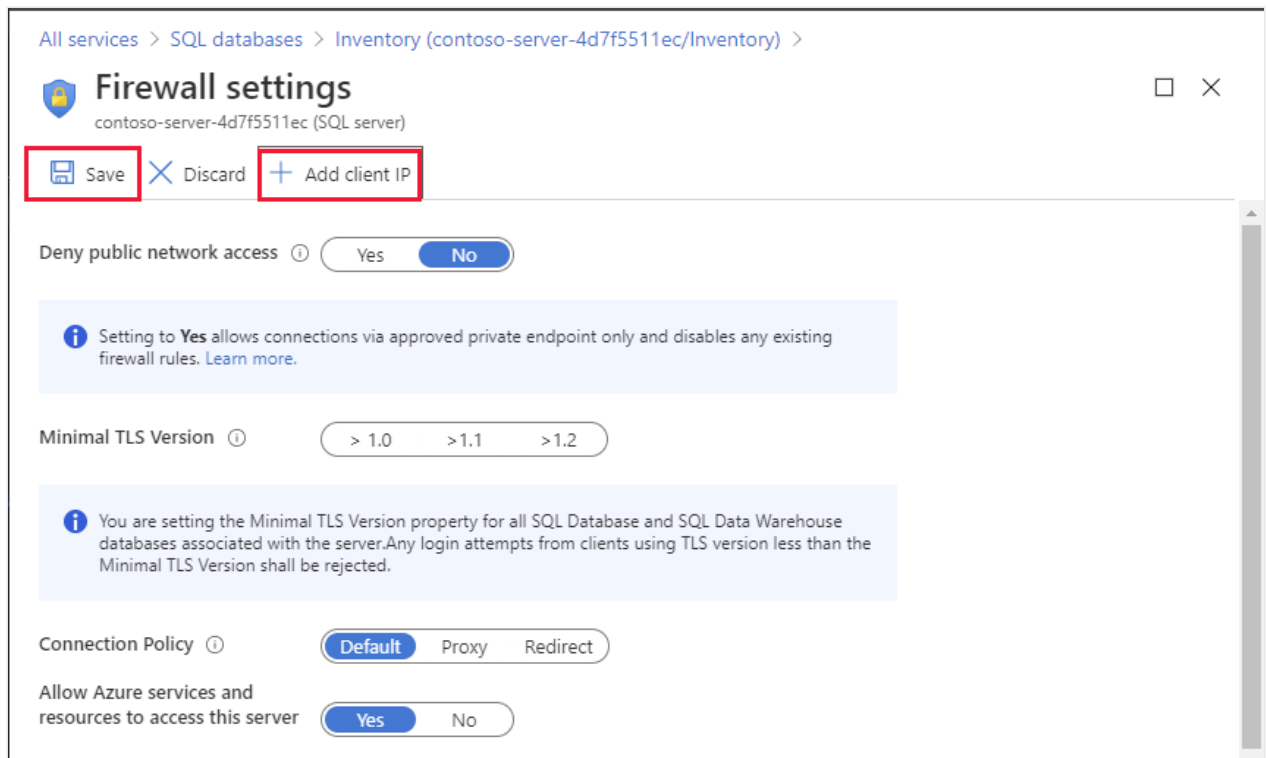
1. Sign into the [Azure portal](#) using the same account you activated the sandbox with.
2. In the portal, on the home page select **SQL databases**, and then select *Inventory* database located on the server you have just created.



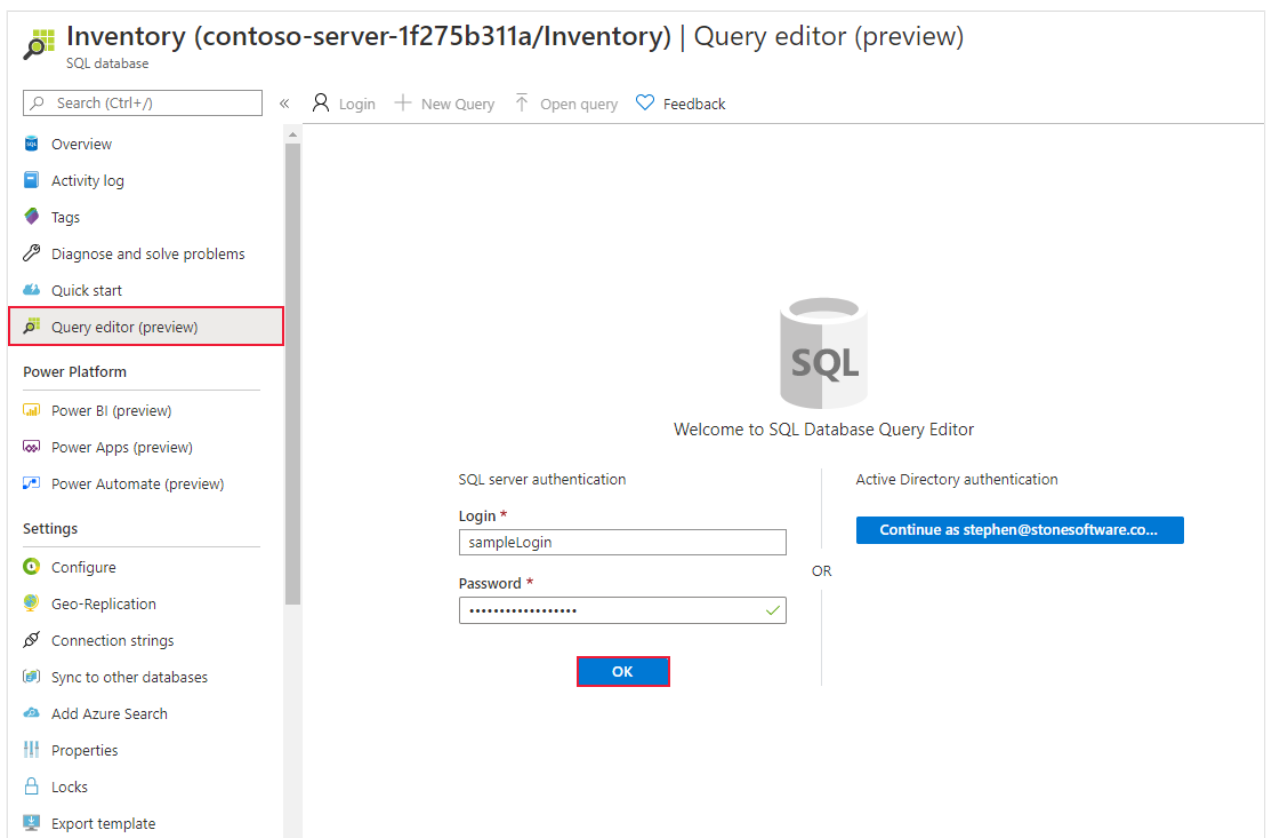
3. On the **Overview** page for your database, select **Set server firewall**.



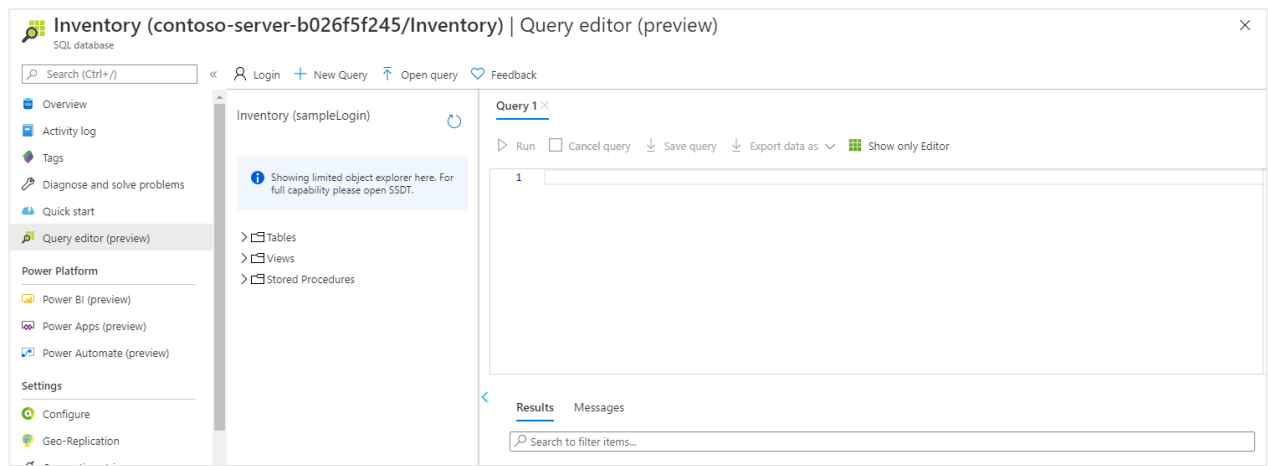
4. On the **Firewall settings** page, select **Add client IP**, and then select **Save**.



5. Close the **Firewall settings** page, and return to the **Overview** page for your database.
6. On the **Overview** page, select **Query editor (preview)** in the left menu.
7. Enter the username and password you recorded earlier when the setup script ran, and then select **OK**.



You'll be presented with a screen similar to this example:

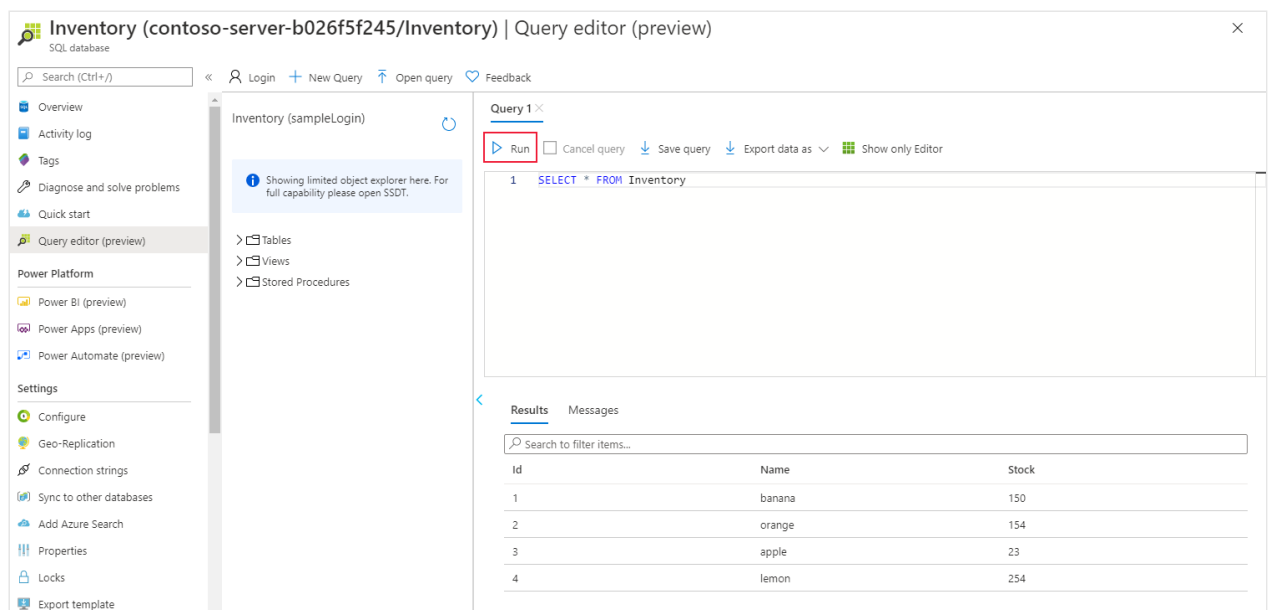
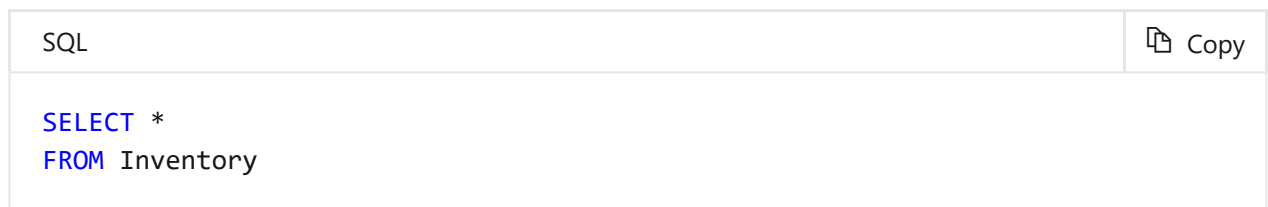


### Tip

Adding your client IP in this step will not account for any existing VPN connections. If you can't complete step 7, disable any VPN connections or add the additional IP address manually from any errors displayed.

## Run queries against the database

1. Copy the following SQL statement into the editor. Select **Run**, to check everything is working. You should see a list of four inventory items



2. Replace the current SQL statement with the following statement to only show the number of bananas in stock:

SQL Copy

```
SELECT *
FROM Inventory
WHERE Name = 'banana'
```

There should be 150 bananas.

The screenshot shows the Azure SQL Database Query Editor interface. The query editor contains the following SQL statement:

```
1 SELECT * FROM Inventory WHERE Name = 'banana'
```

The results pane shows a single row of data:

Id	Name	Stock
1	banana	150

3. Replace the SQL statement with the following statement to retrieve the inventory items in order of the quantity in stock:

SQL Copy

```
SELECT *
FROM Inventory
ORDER BY Stock
```

The screenshot shows the Azure SQL Database Query Editor interface. The query editor contains the following SQL statement:

```
1 SELECT * FROM Inventory ORDER BY Stock
```

The results pane shows a table of inventory items sorted by stock quantity:

Id	Name	Stock
3	apple	23
1	banana	150
2	orange	154
4	lemon	254

4. Replace the SQL statement with the statement shown below. This statement is a query that uses the **JOIN** operator to combine data from the *CustomerOrder* table and the *Inventory* table. It lists the details of orders placed by customers together with the inventory information for each item ordered:

SQL
Copy

```

SELECT *
FROM Inventory
JOIN CustomerOrder ON Inventory.Id = CustomerOrder.InventoryId

```

	Name	Stock		CustomerName	
1	orange	154	1	John Smith	5
2	orange	154	2	Jane Brown	8
3	apple	23	3	Stephen Stone	3
1	banana	150	4	Claire Smith	1
4	lemon	254	5	Sarah Fedun	3
3	apple	23	6	Graham Hinson	9

Query succeeded | 0s

5. Change the query to find the names of all customers who have ordered oranges.

SQL
Copy

```

SELECT CustomerOrder.CustomerName
FROM CustomerOrder
JOIN Inventory ON CustomerOrder.InventoryId = Inventory.ID
AND Inventory.Name = 'orange'

```

This query should return two customers: John Smith and Jane Brown

6. Find out how many customers have ordered lemons. This query uses the **COUNT(\*)** function, which returns the number of rows that match the query criteria.

SQL
Copy


```

SELECT COUNT(*)
FROM CustomerOrder
JOIN Inventory ON CustomerOrder.InventoryId = Inventory.ID
AND Inventory.Name = 'lemon'

```


The results of this query should indicate that only one customer has ordered lemons.

#### 7. Which fruits has John Smith ordered?

SQL	 Copy
<pre>SELECT Inventory.Name FROM CustomerOrder JOIN Inventory ON CustomerOrder.InventoryId = Inventory.ID AND CustomerOrder.CustomerName = 'John Smith'</pre>	

The results of this query should show that John Smith has only ordered oranges.


#### 8. What is the total quantity of items ordered by all customers? The *Quantity* column in the *CustomerOrder* table contains the quantity for each order. This query uses the **SUM** aggregate function to add the quantities together to product a grand total:

SQL	 Copy
<pre>SELECT SUM(CustomerOrder.Quantity) FROM CustomerOrder</pre>	

The answer should be 29.

You've now seen how to run SQL queries against a SQL database. If you have time, try to add some more rows into both tables using **INSERT** statements, modify the rows using **UPDATE** statements, and remove rows using **DELETE** statements.

## Next unit: Summary

[Continue >](#) English (United States) Theme[Manage cookies](#) [Previous Version Docs](#) [Blog](#) [Contribute](#) [Privacy & Cookies](#) [Terms of Use](#)[Trademarks](#) [© Microsoft 2021](#)

 Azure Cloud Shell

