

Lab 3 - SQL&GPT: Deploy Application and test SQL Writing & Data Analysis Assistants

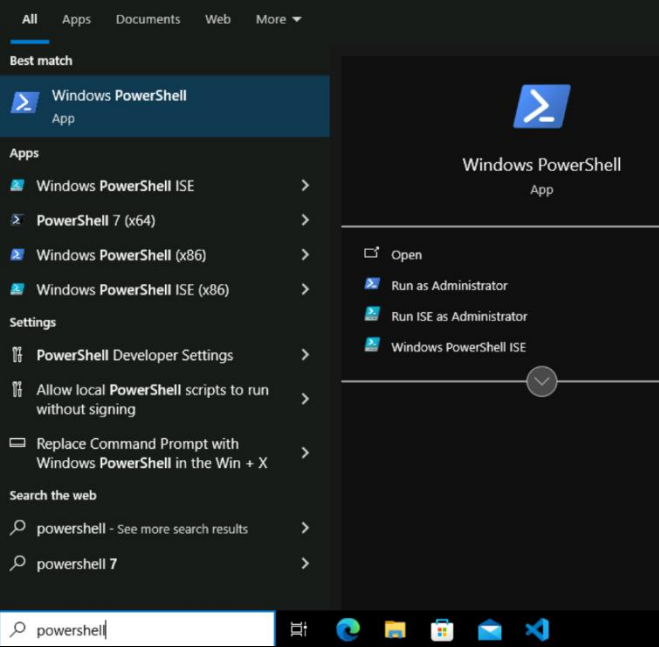
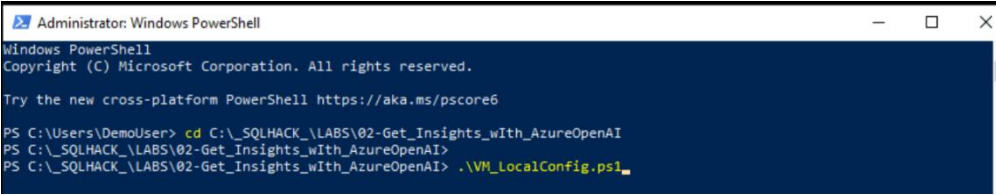
Contents

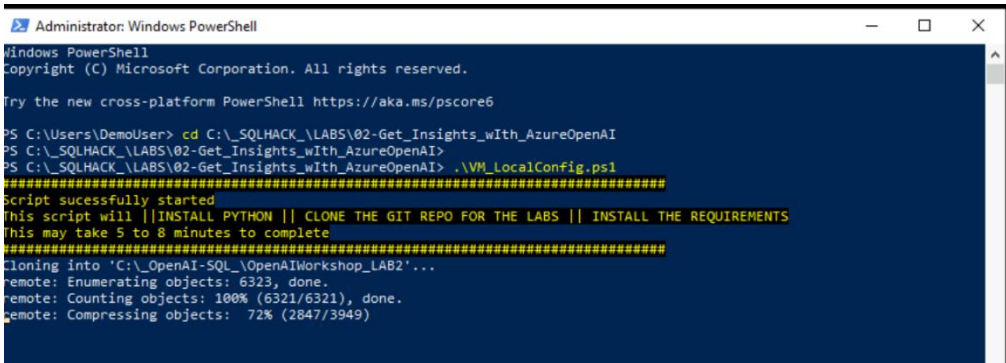
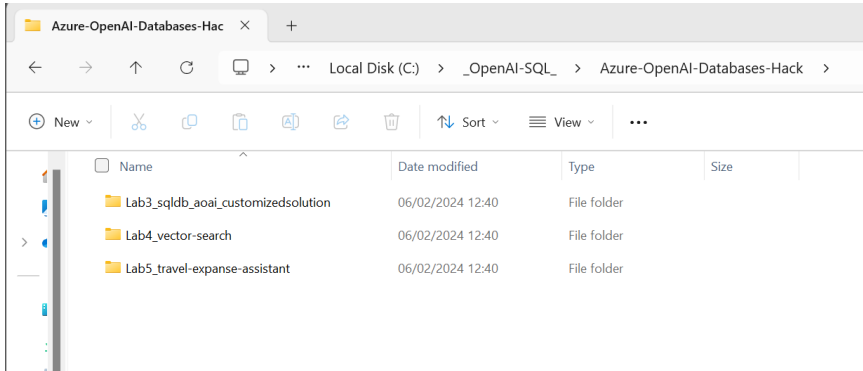
- 1. Install the application locally 2
- 2. Test application and its assistants 8
- 3. Insert new data via natural language..... 16
- 4. Resolve data quality issues 21
- Appendix 32
 - Azure OpenAI Service..... 32
 - Environment Setup 38



1. Install the application locally

The required environment has been provisioned upfront for this workshop. This includes Visual Studio Code, GIT, Python, and an Azure OpenAI resource. If you are interested in setting up the environment on your local machine, you can refer to the “environment setup” section in the appendix.

3. Narrative	Screenshot	Notes
<p>Switch back to your Team VM.</p> <p>Once you connect to your Virtual Machine, start Windows PowerShell from the Windows start menu, make sure you run it as Administrator (right click on Windows PowerShell).</p>		
<p>In PowerShell command line, navigate to following folder with the command:</p> <p><i>cd C:_SQLHACK_\LABS\02-Get_Insights_with_AzureOpenAI</i></p>		

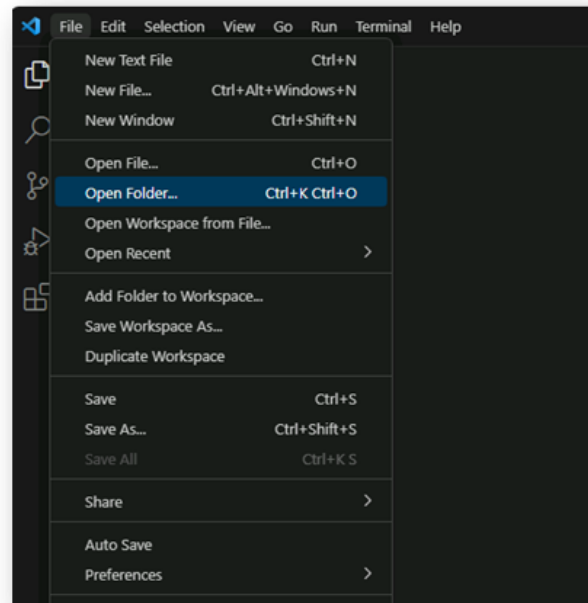
<p>Run the file <i>VM_LocalConfig.ps1</i> by typing in:</p> <p><i>.\VM_LocalConfig.ps1</i></p> <p>Hit enter and let the script run. This might take some time.</p>	 <pre>Administrator: Windows PowerShell Windows PowerShell Copyright (C) Microsoft Corporation. All rights reserved. Try the new cross-platform PowerShell https://aka.ms/pscore6 PS C:\Users\DemoUser> cd C:_SQLHACK_\LABS\02-Get_Insights_wIth_AzureOpenAI PS C:_SQLHACK_\LABS\02-Get_Insights_wIth_AzureOpenAI> PS C:_SQLHACK_\LABS\02-Get_Insights_wIth_AzureOpenAI> .\VM_LocalConfig.ps1 ##### Script successfully started This script will INSTALL PYTHON CLONE THE GIT REPO FOR THE LABS INSTALL THE REQUIREMENTS This may take 5 to 8 minutes to complete ##### Cloning into 'C:_OpenAI-SQL_\OpenAIWorkshop_LAB2'... remote: Enumerating objects: 6323, done. remote: Counting objects: 100% (6321/6321), done. remote: Compressing objects: 72% (2847/3949)</pre>	<p>This script will:</p> <ul style="list-style-type: none">- Install Python- clone the git Repo for the Labs 2 & 3- Install the Python Requirements- Set some environment variables																
<p>The repository with the 3 labs was cloned for you.</p> <p>Check by navigating to path in the file explorer: <i>C:_OpenAI-SQL_\Azure-OpenAI-Databases-Hack</i></p> <p>This is the folder/project you will open in Visual Studio Code.</p>	 <table><thead><tr><th>Name</th><th>Date modified</th><th>Type</th><th>Size</th></tr></thead><tbody><tr><td>Lab3_sqldb_aaoi_customizedsolution</td><td>06/02/2024 12:40</td><td>File folder</td><td></td></tr><tr><td>Lab4_vector-search</td><td>06/02/2024 12:40</td><td>File folder</td><td></td></tr><tr><td>Lab5_travel-expense-assistant</td><td>06/02/2024 12:40</td><td>File folder</td><td></td></tr></tbody></table>	Name	Date modified	Type	Size	Lab3_sqldb_aaoi_customizedsolution	06/02/2024 12:40	File folder		Lab4_vector-search	06/02/2024 12:40	File folder		Lab5_travel-expense-assistant	06/02/2024 12:40	File folder		
Name	Date modified	Type	Size															
Lab3_sqldb_aaoi_customizedsolution	06/02/2024 12:40	File folder																
Lab4_vector-search	06/02/2024 12:40	File folder																
Lab5_travel-expense-assistant	06/02/2024 12:40	File folder																

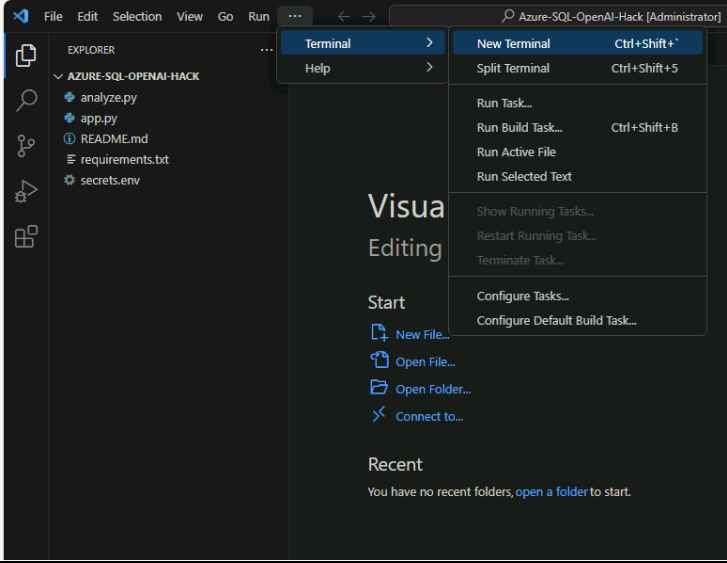
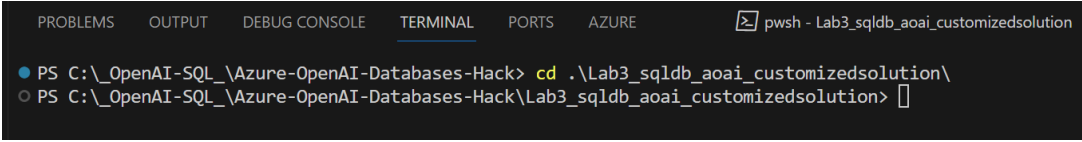
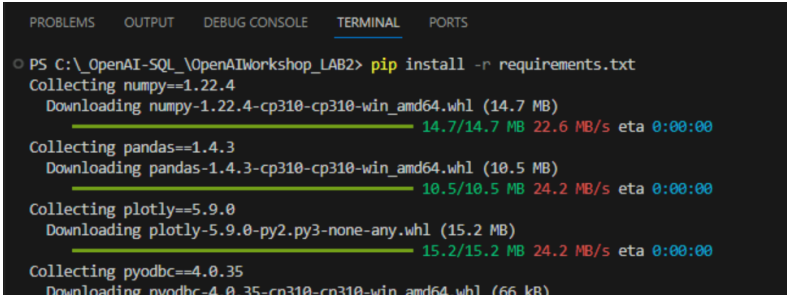
Open **Visual Studio Code** in the Windows start menu.

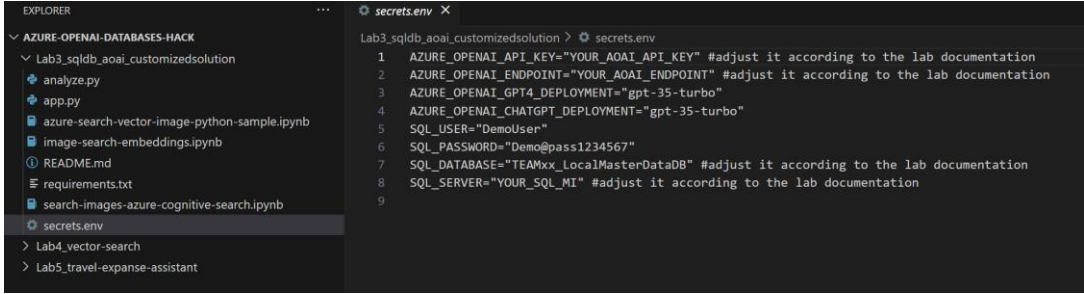
Import the project/folder via **File > Open Folder**.

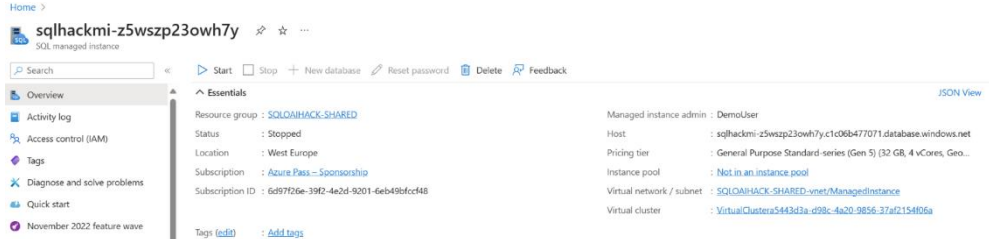

Then select the folder **"C:_OpenAI-SQL_\Azure-OpenAI-Databases-Hack"**

When prompted, check the "Trust the authors of all files" option and click **Yes, I trust the authors**.



<p>Open a terminal via Terminal Tab > New Terminal</p>		
<p>Navigate to the lab 3 subfolder with the cd command:</p> <pre>cd .\Lab3_sqldb_aoi_customizedsolution\</pre>		
<p>Check if the needed requirements are already installed with the command:</p> <pre>pip install -r requirements.txt</pre>		

<p>Now set the credential values within the application code (environment file).</p> <p>Navigate to the secrets.env file with the secrets in Lab3 folder.</p> <p>These should be replaced with the values of the services we use from the Hack subscriptions.</p>																	
<p>Update all your credentials according to the ones on the right side. Make sure you fill in the highlighted values.</p> <p>You can copy the credentials on the right side to your env file.</p>	<pre> AZURE_OPENAI_API_KEY="CHOOSE FROM THE TABLE BELOW" AZURE_OPENAI_ENDPOINT="CHOOSE FROM THE TABLE BELOW" AZURE_OPENAI_GPT4_DEPLOYMENT="gpt-4-32k" AZURE_OPENAI_CHATGPT_DEPLOYMENT="gpt-4-32k" SQL_USER="DemoUser" SQL_PASSWORD="Demo@pass1234567" SQL_DATABASE="TEAMXX_LocalMasterDataDB" SQL_SERVER="sqlhackmi-j754o5hum2r36.7a59bf01d694.database.windows.net" </pre>																
<p>For the Azure OpenAI Endpoint + API Key, use the values according to your team number.</p>	<table border="1"> <thead> <tr> <th>TEAM</th><th>AOAI Endpoint</th><th>AOAI API KEY</th></tr> </thead> <tbody> <tr> <td>Team 01 – 05</td><td>https://sqloai-hack-aue.openai.azure.com/</td><td>6b1afdc3eeab4ae4a294de63a6972c06</td></tr> <tr> <td>Team 06 – 10</td><td>https://sqloai-hack-cae.openai.azure.com/</td><td>d935e90e9a8e496390f41b8ec13c11d6</td></tr> <tr> <td>Team 11 – 15</td><td>https://sqloai-hack-swd.openai.azure.com/</td><td>0c59924d91d24708bdbfd6d1f9333599</td></tr> <tr> <td>Team 16 - 20</td><td>https://sqloai-hack-szn.openai.azure.com/</td><td>20ecb856b5c04c3b89b368abccc82dd5</td></tr> </tbody> </table>	TEAM	AOAI Endpoint	AOAI API KEY	Team 01 – 05	https://sqloai-hack-aue.openai.azure.com/	6b1afdc3eeab4ae4a294de63a6972c06	Team 06 – 10	https://sqloai-hack-cae.openai.azure.com/	d935e90e9a8e496390f41b8ec13c11d6	Team 11 – 15	https://sqloai-hack-swd.openai.azure.com/	0c59924d91d24708bdbfd6d1f9333599	Team 16 - 20	https://sqloai-hack-szn.openai.azure.com/	20ecb856b5c04c3b89b368abccc82dd5	
TEAM	AOAI Endpoint	AOAI API KEY															
Team 01 – 05	https://sqloai-hack-aue.openai.azure.com/	6b1afdc3eeab4ae4a294de63a6972c06															
Team 06 – 10	https://sqloai-hack-cae.openai.azure.com/	d935e90e9a8e496390f41b8ec13c11d6															
Team 11 – 15	https://sqloai-hack-swd.openai.azure.com/	0c59924d91d24708bdbfd6d1f9333599															
Team 16 - 20	https://sqloai-hack-szn.openai.azure.com/	20ecb856b5c04c3b89b368abccc82dd5															

<p>For the SQL_DATABASE value, change it to your team's value, e.g. for Team 01, set it to TEAM01_LocalMasterDataDB.</p> <p>Save the updates (press CTRL+S).</p> <p>The secrets.env file will prepopulate the values in the application UI (user interfaces).</p> <p>Later, you can still change the values using the application's User Interface (UI). This will be done in exercise 3 "Test application and its assistants".</p>	<p>Optionally, you can check the SQL_SERVER value again in the Azure Portal. The SQL_SERVER value is the managed instance host name. Example:</p> 	
<p>In Visual Studio Code, in a terminal, make sure you navigate to the lab 3 folder.</p> <p>To run the application run the command:</p> <p>streamlit run app.py</p>		

2. Test application and its assistants

If prompted to give an **e-mail address**, put your demo user address. E.g. `sqloaihackuserXX@M365x04034558.onmicrosoft.com`

An **Edge browser page** opens with settings retrieved from `secrets.env`. You can also manually change the connection information in the user interface if needed.

On the left side, select the assistant app **"SQL Query writing Assistant"**.

Open **Settings**, and make sure you select **"sqlserver"** as the SQL Engine. Click **Submit** to save the settings.

Keep GPT Model as **"ChatGPT"**.

Select **"Show code"** & **"Show prompt"** to check what happened in the background to return the answer.

Side note:

The option **"sqlite"** is a built-in server by Python for you to test out. In this workshop, we will focus on using our own managed instance (option **"sqlserver"**).

Choose the app

- ☒ SQL Query Writing Assistant
- ☐ Data Analysis Assistant
- ☐ SQL Insert Into Assistant

Settings

Azure OpenAI Settings

ChatGPT deployment name:

gpt-4-32k

GPT-4 deployment name (if not specified, default to ChatGPT's):

gpt-4-32k

Azure OpenAI Endpoint:

https://sqloai-hack-swd.openai.azure.com

Azure OpenAI Key:

SQL Engine:

- ☐ sqlite
- ☒ sqlserver

SQL Server Settings (Optional)

SQL Server:

sqlhackmi-j754o5hum2r36.7a59bft

Database:

TEAM01_LocalMasterDataDB

User:

DemoUser

Password:

Submit

Let's test the **SQL Query writing Assistant**. Make sure you have selected this assistant.

Under “**Ask me a question**” below, you can try out your first sample queries in natural language.

Past the sample 1 below and hit the **Submit** button after.

Sample 1: Show me revenue by product in ascending order

The screenshot shows a web application titled "Natural Language Query" running on localhost:8501. The interface is divided into two main sections. On the left, there's a sidebar with options to "Choose the app" (SQL Query Writing Assistant is selected), "Settings", "GPT Model" (set to ChatGPT), and "FAQs" (with a dropdown menu showing "Show me revenue by product in ascending order"). Below these are checkboxes for "Show code" and "Show prompt", and a section "Ask me a question" with a text input field containing the query "Show me revenue by product in ascending order" and a "Submit" button. On the right, the results of the query are displayed under the heading "Question: Show me revenue by product in ascending order". The results are shown in a table with two columns: "ProductName" and "Revenue".

	ProductName	Revenue
0	LL Road Seat/Saddle	162.72
1	Mountain Bike Socks, L	513
2	LL Touring Frame - Blue, 58	800.208
3	LL Mountain Frame - Black, 40	1,198.992
4	LL Touring Seat/Saddle	1,480.752
5	ML Mountain Frame-W - Silver, 38	1,529.178
6	LL Touring Handlebars	1,548.624
7	LL Headset	1,949.4
8	ML Touring Seat/Saddle	1,972.656
9	LL Mountain Frame - Black, 52	2,248.11

Sample 2: Show me top 10 most expensive products

The screenshot shows a web application titled "Natural Language Query" running on localhost:8501. The interface is divided into a configuration panel on the left and a results panel on the right.

Configuration Panel (Left):

- Choose the app:** Radio buttons for "SQL Query Writing Assistant" (selected) and "Data Analysis Assistant".
- Settings:** A button to open settings.
- GPT Model:** A dropdown menu currently showing "ChatGPT".
- FAQs:** A dropdown menu showing "Show me top 10 most expensive products".
- Options:** Checkboxes for "Show code" (checked) and "Show prompt" (unchecked).
- Ask me a question:** A text input field containing "Show me top 10 most expensive products".
- Submit:** A red button at the bottom of the configuration panel.

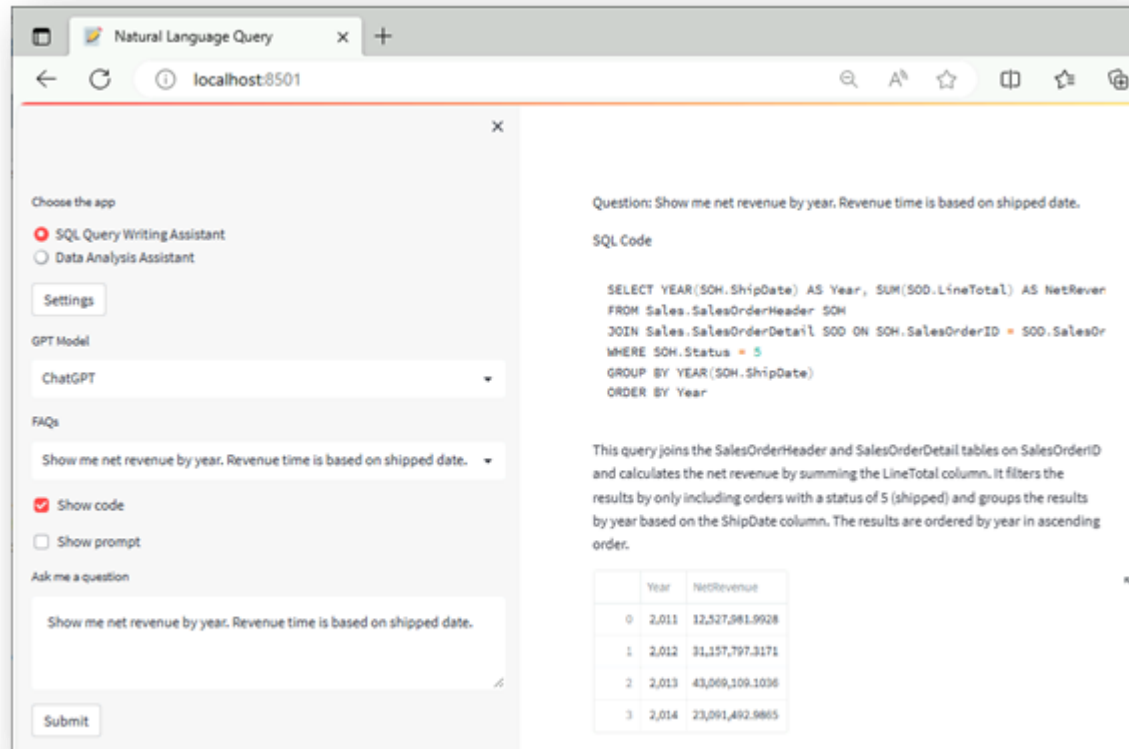
Results Panel (Right):

- Question:** "Show me top 10 most expensive products"
- SQL Code:**

```
SELECT TOP 10 Name, ListPrice
FROM Production.Product
ORDER BY ListPrice DESC
```
- Table:** A table with 3 columns: an index, Name, and ListPrice. It contains 10 rows of product data.

	Name	ListPrice
0	Road-150 Red, 62	3,578.27
1	Road-150 Red, 44	3,578.27
2	Road-150 Red, 48	3,578.27
3	Road-150 Red, 52	3,578.27
4	Road-150 Red, 56	3,578.27
5	Mountain-100 Silver, 38	3,399.99
6	Mountain-100 Silver, 42	3,399.99
7	Mountain-100 Silver, 44	3,399.99
8	Mountain-100 Silver, 48	3,399.99
9	Mountain-100 Black, 38	3,374.99

***Sample 3: Show me net revenue by year.
Revenue time is based on shipped date.***



The screenshot shows a web application titled "Natural Language Query" running on localhost:8501. The interface is divided into a left sidebar and a main content area.

Left Sidebar:

- Choose the app:** Two radio buttons are present: "SQL Query Writing Assistant" (selected) and "Data Analysis Assistant".
- Settings:** A button labeled "Settings".
- GPT Model:** A dropdown menu currently showing "ChatGPT".
- FAQs:** A dropdown menu showing "Show me net revenue by year. Revenue time is based on shipped date.".
- Show code:** A checked checkbox.
- Show prompt:** An unchecked checkbox.
- Ask me a question:** A text input field containing the query "Show me net revenue by year. Revenue time is based on shipped date.".
- Submit:** A button at the bottom of the sidebar.

Main Content Area:

- Question:** "Show me net revenue by year. Revenue time is based on shipped date."
- SQL Code:**

```
SELECT YEAR(SOH.ShipDate) AS Year, SUM(SOD.LineTotal) AS NetRevenue
FROM Sales.SalesOrderHeader SOH
JOIN Sales.SalesOrderDetail SOD ON SOH.SalesOrderID = SOD.SalesOrderID
WHERE SOH.Status = 5
GROUP BY YEAR(SOH.ShipDate)
ORDER BY Year
```
- Explanation:** "This query joins the SalesOrderHeader and SalesOrderDetail tables on SalesOrderID and calculates the net revenue by summing the LineTotal column. It filters the results by only including orders with a status of 5 (shipped) and groups the results by year based on the ShipDate column. The results are ordered by year in ascending order."
- Results Table:**

	Year	NetRevenue
0	2,011	12,527,981.9928
1	2,012	31,157,797.3171
2	2,013	43,069,109.1036
3	2,014	23,091,492.9865

Sample 4: For each category, get the list of products sold and the total sales amount

The screenshot shows a web application titled "Natural Language Query" running on localhost:8501. The interface is divided into two main sections. On the left, there is a sidebar with options to "Choose the app" (SQL Query Writing Assistant is selected), "Settings", "GPT Model" (set to ChatGPT), and "FAQs". Below the FAQs, there is a text input field containing the question "For each category, get the list of products sold and the total sales amount", a checkbox for "Show code" which is checked, and a "Submit" button. On the right, the main content area displays the question, the generated SQL code, and a table of results. The SQL code is a complex query joining Production.Product, Production.ProductSubcategory, and Sales.SalesOrderDetail tables. The table of results has three columns: CategoryName, ProductName, and TotalSalesAmount, and lists 10 rows of data for Mountain Bikes and Road Bikes.

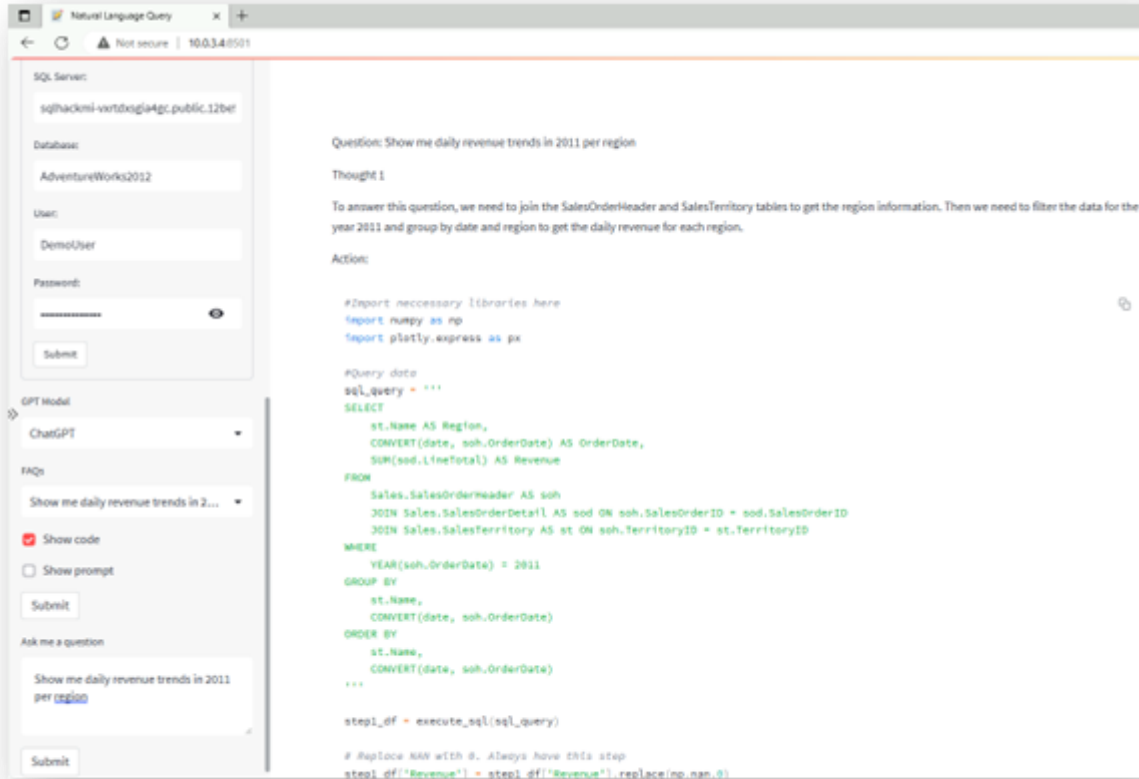
Question: For each category, get the list of products sold and the total sales amount

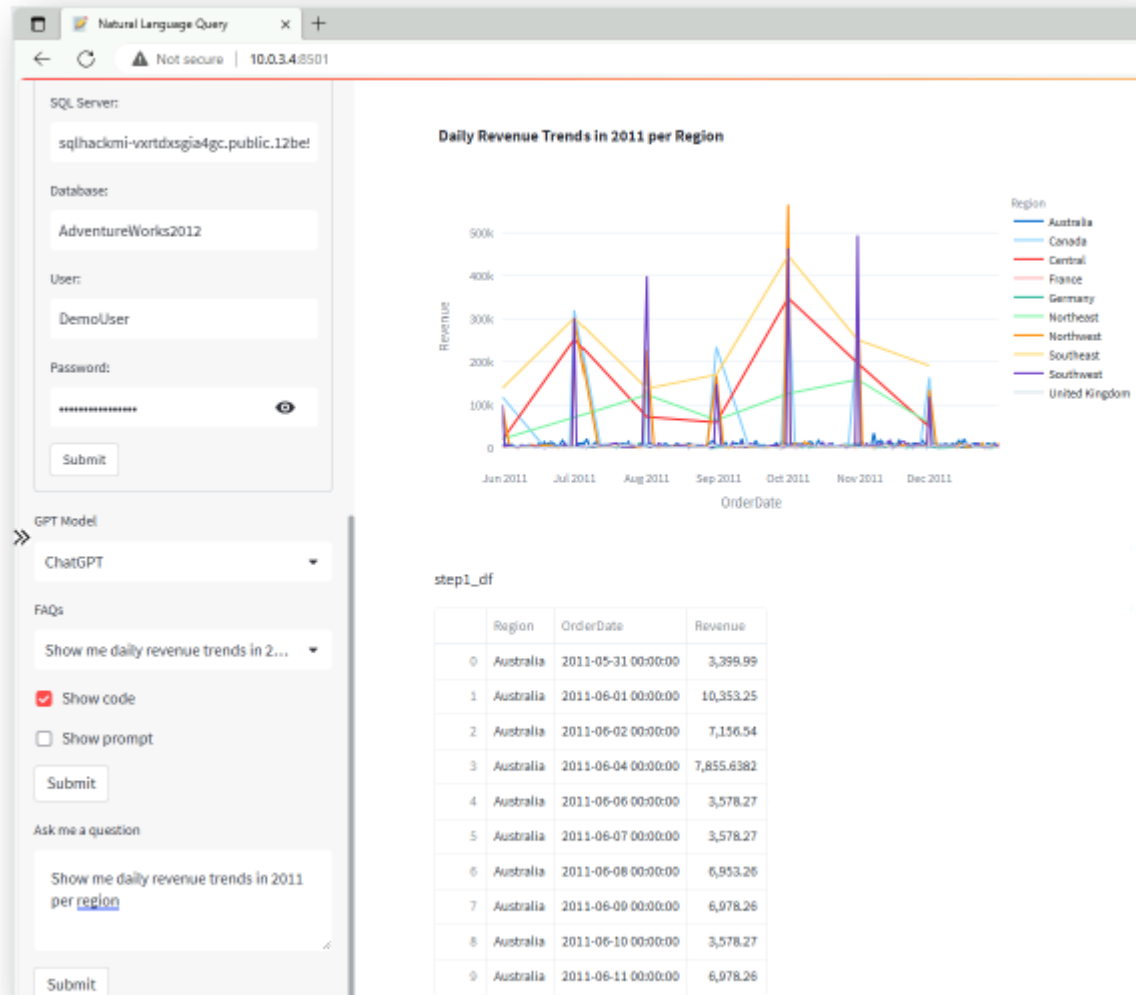
SQL Code

```
SELECT pc.Name AS CategoryName, p.Name AS ProductName, SUM(sod.Li
FROM Production.Product p
JOIN Production.ProductSubcategory pc ON p.ProductSubcategoryID =
JOIN Sales.SalesOrderDetail sod ON p.ProductID = sod.ProductID
GROUP BY pc.Name, p.Name
ORDER BY TotalSalesAmount DESC
```

	CategoryName	ProductName	TotalSalesAmount
0	Mountain Bikes	Mountain-200 Black, 38	4,400,592.8004
1	Mountain Bikes	Mountain-200 Black, 42	4,009,494.7618
2	Mountain Bikes	Mountain-200 Silver, 38	3,693,678.0253
3	Mountain Bikes	Mountain-200 Silver, 42	3,438,478.8604
4	Mountain Bikes	Mountain-200 Silver, 46	3,434,256.9419
5	Mountain Bikes	Mountain-200 Black, 46	3,309,673.2169
6	Road Bikes	Road-250 Black, 44	2,516,857.3149
7	Road Bikes	Road-250 Black, 48	2,347,655.9535
8	Road Bikes	Road-250 Black, 52	2,012,447.775
9	Road Bikes	Road-150 Red, 56	1,847,818.628

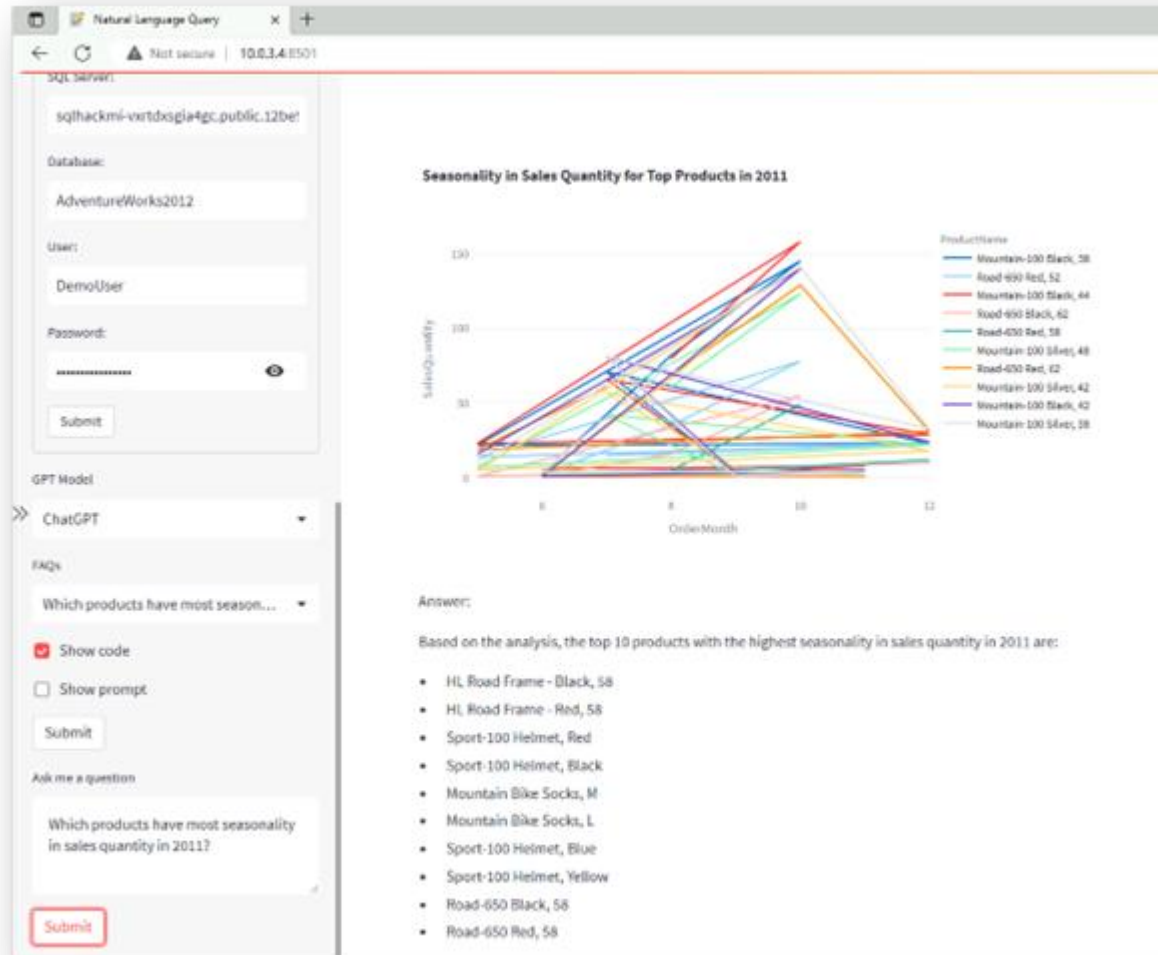
Now, let's test the second assistant "Data Analysis Assistant" to generate further insights – also using natural language.

Narrative	Screenshot	Notes
<p>In the UI, switch to the "Data Analysis Assistant".</p> <p>Make sure to select "sqlserver" as SQL Engine under connecting settings. Then hit submit.</p> <p>Now, you can paste the following query and hit submit:</p> <p>Sample 1: Show me daily revenue trends in 2011 per region</p>	 <p>The screenshot shows the 'Natural Language Query' web application. On the left, the 'SQL Server' configuration is set to 'sqlhackmi-virtxigla4gc.public.12b.e', the database is 'AdventureWorks2012', and the user is 'DemoUser'. The 'GPT Model' is set to 'ChatGPT'. The 'FAQs' section shows the question 'Show me daily revenue trends in 2011 per region'. The 'Show code' checkbox is checked, displaying the following SQL query:</p> <pre> -- Import necessary libraries here report nuppy as np report plotly.express as px #Query data sql_query = """ SELECT st.Name AS Region, CONVERT(date, soh.OrderDate) AS OrderDate, SUM(sod.LineTotal) AS Revenue FROM Sales.SalesOrderHeader AS soh JOIN Sales.SalesOrderDetail AS sod ON soh.SalesOrderID = sod.SalesOrderID JOIN Sales.SalesTerritory AS st ON soh.TerritoryID = st.TerritoryID WHERE YEAR(soh.OrderDate) = 2011 GROUP BY st.Name, CONVERT(date, soh.OrderDate) ORDER BY st.Name, CONVERT(date, soh.OrderDate) """ step1_df = execute_sql(sql_query) # Replace NaN with 0. Always have this step step1_df['Revenue'] = step1_df['Revenue'].replace(np.nan, 0) </pre>	



Sample 2:

Which products have most seasonality in sales quantity in 2011?



3. Insert new data via natural language

The task is dedicated to presentation of diverse ways how the initial logic of communication with your SQL DB data might be extended. For this Hackathon, we have worked with INSERT function (flow is presented below). It is worth mentioning that the function was implemented as a trial by Hackathon presenters.

Use Case Scenario: Recording Scrap Data in LocalMasterDataDB.

Background: LocalMasterDataDB is a manufacturing plant that produces a variety of products. To maintain quality control and minimize waste, the plant has a Scrap table in its database to record and analyze scrapped parts and materials. (Note: there are special systems that collect information about the scrapped parts on the production lines at plants. The presented Use Case is just an example of insert functionality).

Scenario: In the LocalMasterDataDB manufacturing plant, the Scrap Redemption Center plays a crucial role in documenting and managing scrap data. To provide information about the reason of part rejection, the worker enters information about the type of scrap (e.g., "Gouge in metal," "Drill size too large").

Narrative	Screenshot	Notes																												
<p>Select the SQL Query Writing Assistant. Make sure in settings, sqlserver is selected as SQL Engine. Click Submit to save the settings.</p> <p>Check the reasons for the scrap parts with the following prompt:</p> <p><i>Show me the possible reasons of Scrap Parts</i></p>	<div><p>Question: show me the possible reasons of scrap from Scrap Table</p><table><thead><tr><th></th><th>Name</th></tr></thead><tbody><tr><td>0</td><td>Brake assembly not as ordered</td></tr><tr><td>1</td><td>Color incorrect</td></tr><tr><td>2</td><td>Drill pattern incorrect</td></tr><tr><td>3</td><td>Drill size too large</td></tr><tr><td>4</td><td>Drill size too small</td></tr><tr><td>5</td><td>Gouge in metal</td></tr><tr><td>6</td><td>Handling damage</td></tr><tr><td>7</td><td>Paint process failed</td></tr><tr><td>8</td><td>Primer process failed</td></tr><tr><td>9</td><td>Seat assembly not as ordered</td></tr><tr><td>10</td><td>Stress test failed</td></tr><tr><td>11</td><td>Thermoform temperature too high</td></tr><tr><td>12</td><td>Thermoform temperature too low</td></tr></tbody></table></div>		Name	0	Brake assembly not as ordered	1	Color incorrect	2	Drill pattern incorrect	3	Drill size too large	4	Drill size too small	5	Gouge in metal	6	Handling damage	7	Paint process failed	8	Primer process failed	9	Seat assembly not as ordered	10	Stress test failed	11	Thermoform temperature too high	12	Thermoform temperature too low	
	Name																													
0	Brake assembly not as ordered																													
1	Color incorrect																													
2	Drill pattern incorrect																													
3	Drill size too large																													
4	Drill size too small																													
5	Gouge in metal																													
6	Handling damage																													
7	Paint process failed																													
8	Primer process failed																													
9	Seat assembly not as ordered																													
10	Stress test failed																													
11	Thermoform temperature too high																													
12	Thermoform temperature too low																													

Insert with the Staging Table

Narrative	Screenshot/Code	Notes
<p>Back in SSMS (SQL Server Management Studio), open a query window within your team LocalMasterDataDB database (e.g. TEAM01_LocalMasterDataDB). You can do so by right clicking on the DB and selecting “New Query”.</p> <p>Make sure you are in the right context (your Team DB) before executing the query.</p> <p>Create a staging table with the T-SQL statement on the right and insert your TeamXX name for the table.</p>	<pre>CREATE TABLE [TeamXX_StagingTable] (ScrapID INT IDENTITY(1,1) PRIMARY KEY, -- Auto-incremented unique identifier for each scrap record ProductID INT, -- Foreign key to identify the product associated with the scrap ScrappedQty INT, -- Quantity of scrapped parts ScrapReasonID INT, -- Foreign key to specify the reason for scrap ScrapDate DATETIME DEFAULT GETDATE() -- Date and time of scrap (default to current timestamp));</pre>	

Check if the table is already there with the help of **SQL Query Writing Assistant**.

Use the prompt:

Show me if there is any staging table

The screenshot shows the SQL Query Writing Assistant interface. On the left, under 'Choose the app', 'SQL Query Writing Assistant' is selected. Below it, 'GPT Model' is set to 'ChatGPT'. Under 'FAQs', 'Show me revenue by product in asc...' is selected. There are checkboxes for 'Show code' (checked) and 'Show prompt' (unchecked). At the bottom, the 'Ask me a question' section contains the prompt 'Show me if there is any staging table'. On the right, the 'Question' is 'Show me if there is any staging table'. Below it, the 'SQL Code' is displayed as 'SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME LIKE'. At the bottom, a table shows the results of the query:

	TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	TABLI
0	TEAM18_LocalMasterDataDB	dbo	Team18_StagingTable	BASE

Now we are fine to add our first information about the scrap parts (add your values here).

Switch to **SQL Insert Into Assistant**.

Make sure to select “**sqlserver**” as SQL Engine in the Settings and hit **submit**.

Now paste this prompt (adjust your staging table name):

Insert a new item to **TeamXX_StagingTable** where *productID* is 1, *scrap quantity* is 4 and *scrap reason* is 7.

Choose the app

☐ SQL Query Writing Assistant
☐ Data Analysis Assistant
☒ SQL Insert Into Assistant

Settings

GPT Model

ChatGPT

FAQs

Show me revenue by product in asc...

☒ Show code
☐ Show prompt

Ask me a question

This query will insert a new item to the Team18_StagingTable with ProductID of 1, ScrappedQty of 4, ScrapReasonID of 7, and ScrapDate of the current date and time.

Submit

Question: This query will insert a new item to the Team18_StagingTable with ProductID of 1, ScrappedQty of 4, ScrapReasonID of 7, and ScrapDate of the current date and time.

SQL Code

```
INSERT INTO Team18_StagingTable (ProductID, ScrappedQty, ScrapReasonID, ScrapDate)
VALUES (1, 4, 7, GETDATE())
```

Your insert was successful to TEAM18_LocalMasterDataDB. Please check the database to verify.

Insert a new value and check that the primary key (ScrapID) is automatically increased with the following prompt (again, adjust to your staging table name:

*Insert a new item to the **TeamXX_StagingTable** with ScrappedQty of 2, ScrapReasonID of 9, and ScrapDate of the current date and time.*

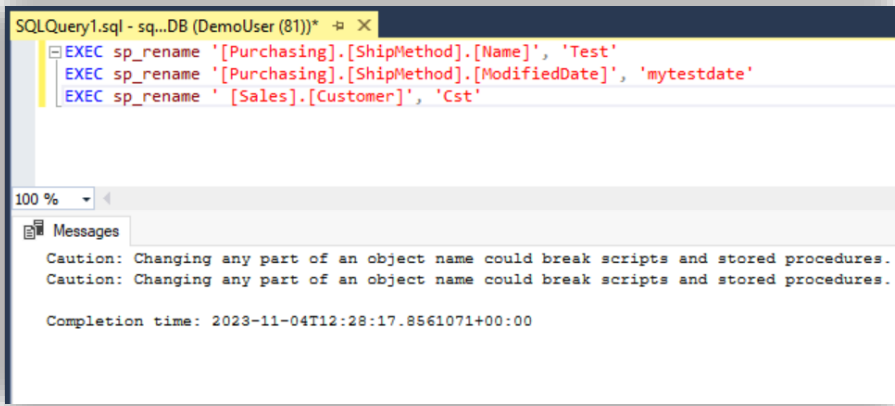
Verify in **SSMS** or in the **SQL Server Writing Assistant** that the item was inserted and there was an automatic increase of ScrapID.

The screenshot shows the SQL Query Writing Assistant interface. On the left, there's a sidebar with options: 'Choose the app' (SQL Query Writing Assistant selected), 'Settings', 'GPT Model' (ChatGPT selected), 'FAQs' (Show me revenue by product in asc... selected), and 'Show code' (checked). The main area displays the question 'show me team18 staging table' and the generated SQL code: 'SELECT * FROM dbo.Team18_StagingTable'. Below the code, a table of results is shown:

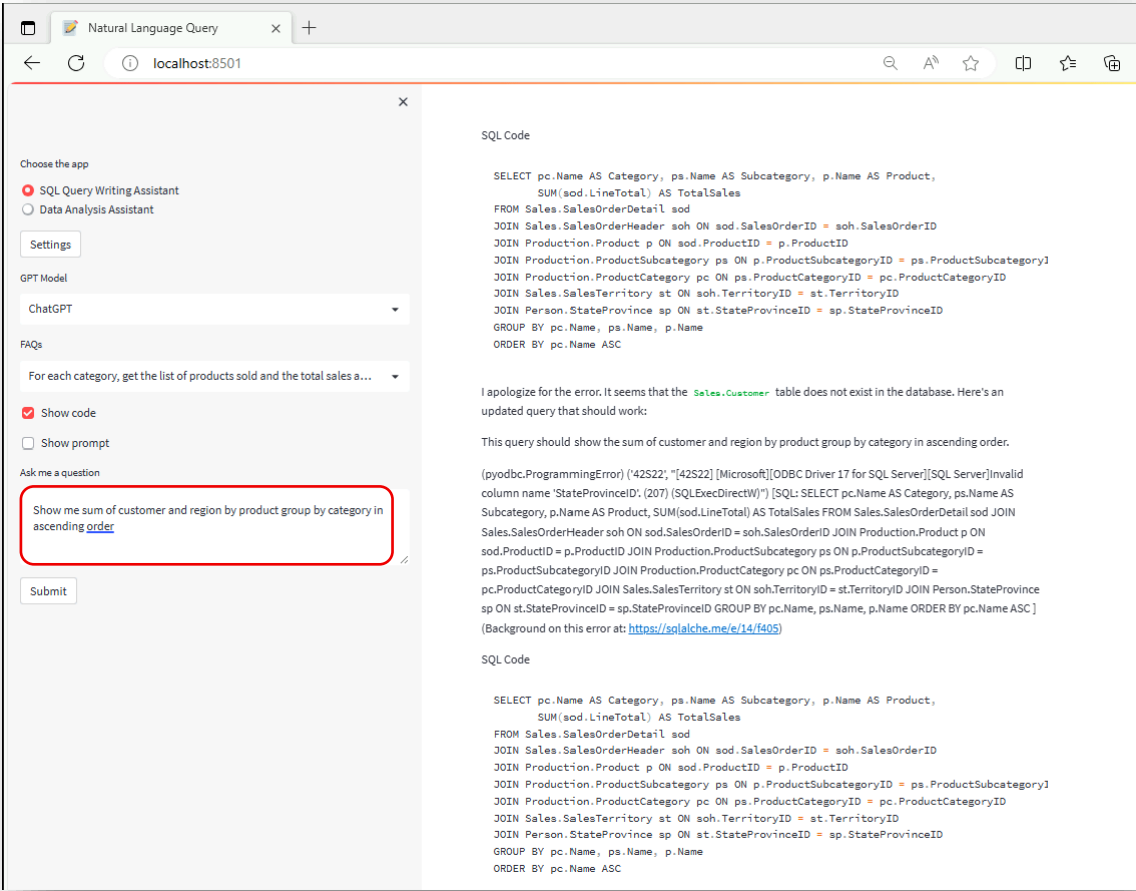
	ScrapID	ProductID	ScrappedQty	ScrapReasonID	ScrapDate
0	1	1	4	7	2023-11-04 13:09:52
1	2	None	2	9	2023-11-04 13:11:02

4. Resolve data quality issues

The purpose of this exercise is to understand the importance of ensuring the data which Azure OpenAI works with has the expected quality. First, let us set up the data quality issues by changing object names in our database. We will then analyse the data quality by checking the database object and naming.

Narrative	Screenshot	Notes
<p>First, let's intentionally create "quality issues within our DB.</p> <p>In SSMS, open a query window on your TeamXX_LocalMasterDataDB and execute the following T-SQL statement:</p> <pre>EXEC sp_rename '[Purchasing].[ShipMethod].[Name]', 'Test' EXEC sp_rename '[Purchasing].[ShipMethod].[ModifiedDate]', 'mytestdate' EXEC sp_rename '[Sales].[Customer]', 'Cst'</pre>		

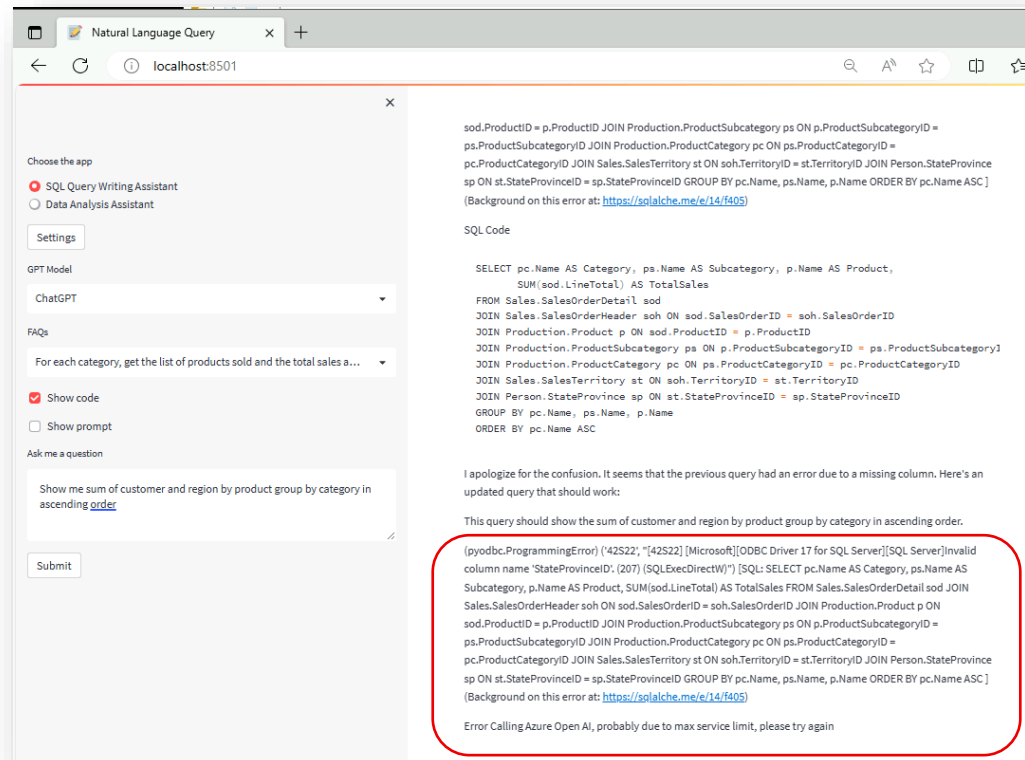
Now, let's try some samples which will run into errors to understand potential data quality issues we would have to resolve.

Narrative	Screenshot	Notes
<p><u>Sample 1:</u></p> <p>Copy the text in the question box, still using SQL Query Assistant:</p> <p><i>Show me customer by product group by category in ascending order</i></p>	 <p>The screenshot shows a web browser window titled 'Natural Language Query' at 'localhost:8501'. The interface has a sidebar on the left with options: 'Choose the app' (SQL Query Writing Assistant selected), 'Settings', 'GPT Model' (ChatGPT), and 'FAQs'. The main area displays a SQL query and an error message. The query is a complex JOIN statement. The error message states: 'I apologize for the error. It seems that the Sales.Customer table does not exist in the database. Here's an updated query that should work:'. Below the error, it says 'This query should show the sum of customer and region by product group by category in ascending order.' and provides a corrected SQL query. The user's input in the 'Ask me a question' box is highlighted with a red rectangle: 'Show me sum of customer and region by product group by category in ascending order'.</p>	

Notice an **error** is thrown.

You may get a similar message such as “*I apologize for the confusion. It seems there was an error in the previous query. The 'Sales.Customer' table does not exist in the database. Instead, we have a 'Sales.Cst' table that contains the 'CustomerID'.*”.

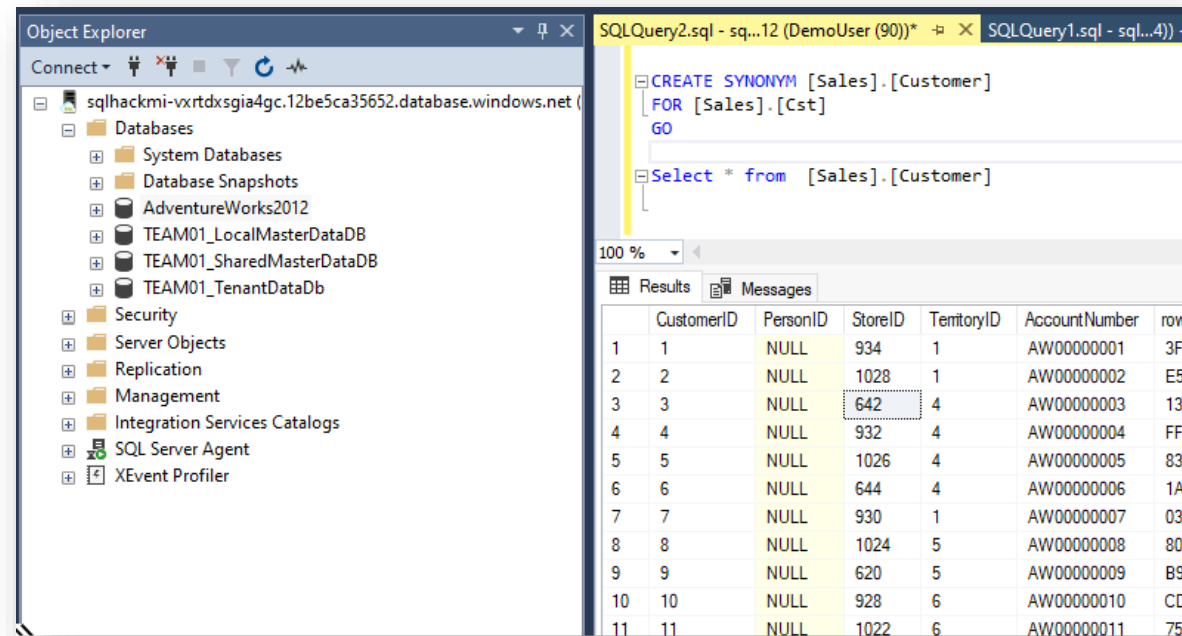
The reason is due to the **object naming** in the database. We purposely changed [Sales].Customer to [Sales].CST. The Azure OpenAI model converts the user query to T-SQL (Transact-SQL). However, since no table with the name “Customer” exists, it cannot query from the table.



<p>You can also test in the Data Analysis Assistant:</p> <p><i>Which customers are most likely to churn?</i></p> <p>This will also throw an error since it also refers to the customer table.</p>	<div data-bbox="703 193 1709 427"> <p>Error:</p> <pre>(pyodbc.ProgrammingError) ('42S02', "[42S02] [Microsoft][ODBC Driver 17 for SQL Server][SQL Server]Invalid object name 'Sales.Customer'. (208) (SQLExecDirectW)") [SQL: SELECT c.CustomerID, c.PersonID, c.StoreID, c.TerritoryID, c.AccountNumber, c.rowguid, c.ModifiedDate FROM Sales.SalesOrderHeader AS o JOIN Sales.Customer AS c ON o.CustomerID = c.CustomerID WHERE o.Status = 5] (Background on this error at: https://sqlalche.me/e/14/f405)</pre> </div>	
<p><u>FIX:</u> In SSMS, create a synonym "sales.customer" for the "sales.cst" table.</p> <p>Within your TEAMXX_LocalMasterDataDB, open a new query window and run the following T-SQL statement:</p> <pre>CREATE SYNONYM [Sales].[Customer] FOR [Sales].[Cst] GO</pre> <p>Review the newly created synonym as shown in the screenshot.</p>	<div data-bbox="629 560 1397 1110"> <p>The screenshot shows the 'Object Explorer' window in SQL Server Enterprise Manager. It is connected to a server named 'sqlhackmi-vxrt dxsgia4gc.12be5ca35652.database.windows.net'. The 'Databases' folder is expanded, showing 'System Databases', 'Database Snapshots', and 'AdventureWorks2012'. Under 'AdventureWorks2012', the 'Synonyms' folder is expanded and highlighted with a red box, showing a single synonym named 'Sales.Customer'.</p> </div>	<p>Learn more about synonyms in our documentation: CREATE SYNONYM (Transact-SQL) - SQL Server Microsoft Learn</p>

Execute the T-sql query::

*Select * from
[Sales].[Customer]*



The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure, including System Databases, Database Snapshots, AdventureWorks2012, and TEAM01 databases. The SQL Query window on the right shows the execution of a T-SQL query. The query is as follows:

```
CREATE SYNONYM [Sales].[Customer]
FOR [Sales].[Cst]
GO
Select * from [Sales].[Customer]
```

The Results tab shows the output of the query, which is a table with 11 rows and 6 columns: CustomerID, PersonID, StoreID, TerritoryID, AccountNumber, and rowid. The data is as follows:

	CustomerID	PersonID	StoreID	TerritoryID	AccountNumber	rowid
1	1	NULL	934	1	AW00000001	3F5
2	2	NULL	1028	1	AW00000002	E5
3	3	NULL	642	4	AW00000003	130
4	4	NULL	932	4	AW00000004	FF
5	5	NULL	1026	4	AW00000005	83
6	6	NULL	644	4	AW00000006	1A
7	7	NULL	930	1	AW00000007	03
8	8	NULL	1024	5	AW00000008	80
9	9	NULL	620	5	AW00000009	B9
10	10	NULL	928	6	AW00000010	CD
11	11	NULL	1022	6	AW00000011	75

**Submit the question again
and view the result:**

Natural Language Query

localhost:8501

Choose the app

☒ SQL Query Writing Assistant
 ☐ Data Analysis Assistant

Settings

GPT Model

ChatGPT

FAQs

For each category, get the list of products sold and the total sales a...

☒ Show code
 ☐ Show prompt

Ask me a question

Show me sum of customer and region by product group by category in ascending order

Submit

Question: Show me sum of customer and region by product group by category in ascending order

SQL Code

```

SELECT pc.Name AS Category, p.ProductSubcategoryID, ps.Name AS S
p.ProductID, p.Name AS Product,
SUM(sod.OrderQty) AS TotalQuantity,
SUM(sod.LineTotal) AS TotalSales
FROM Sales.SalesOrderDetail sod
JOIN Production.Product p ON sod.ProductID = p.ProductID
JOIN Production.ProductSubcategory ps ON p.ProductSubcategoryID
JOIN Production.ProductCategory pc ON ps.ProductCategoryID = pc.
JOIN Sales.SalesOrderHeader soh ON sod.SalesOrderID = soh.SalesO
JOIN Sales.Customer c ON soh.CustomerID = c.CustomerID
JOIN Person.BusinessEntityAddress bea ON c.PersonID = bea.Busine
JOIN Person.Address a ON bea.AddressID = a.AddressID
JOIN Person.StateProvince sp ON a.StateProvinceID = sp.StateProv
JOIN Sales.SalesTerritory st ON c.TerritoryID = st.TerritoryID
GROUP BY pc.Name, p.ProductSubcategoryID, ps.Name, p.ProductID,
ORDER BY pc.Name ASC

```

	Category	ProductSubcategoryID	Subcategory	ProductID	Product
0	Accessories	28	Bottles and Cages	872	Road Bottl
1	Accessories	37	Tires and Tubes	922	Road Tire
2	Accessories	30	Fenders	878	Fender Set
3	Accessories	37	Tires and Tubes	931	LL Road Ti
4	Accessories	31	Helmets	707	Sport-100
5	Accessories	37	Tires and Tubes	934	Touring Ti
6	Accessories	37	Tires and Tubes	928	LL Mounta
7	Accessories	32	Hydration Packs	880	Hydration
8	Accessories	37	Tires and Tubes	929	ML Mount
9	Accessories	27	Bike Stands	879	All-Purpos

Sample 2

Copy the text in the question box:

Find Quarterly Orders by Product. First column is Product Name, then year then four other columns, each for a quarter group by ship method

The screenshot shows the Natural Language Query interface. On the left, there's a sidebar with options like 'Choose the app' (SQL Query Writing Assistant selected), 'Settings', 'GPT Model' (ChatGPT selected), and 'FAQs'. Below that, there's a 'Find Quarterly Orders by Product. F...' dropdown and checkboxes for 'Show code' and 'Show prompt'. At the bottom, there's a 'Submit' button.

The main area displays a SQL query:

```
pm.Name AS ShipMethod
FROM
  Production.Product p
  JOIN Purchasing.PurchaseOrderDetail pod ON p.ProductID = pod.ProductID
  JOIN Purchasing.PurchaseOrderHeader po ON pod.PurchaseOrderID = po.PurchaseOrderID
  JOIN Purchasing.ShipMethod pm ON po.ShipMethodID = pm.ShipMethodID
GROUP BY
  p.Name,
  YEAR(po.OrderDate),
  pm.Name
ORDER BY
  p.Name,
  YEAR(po.OrderDate),
  pm.Name
```

Below the query, there's an error message:

It looks like there is an error in the query. The error message suggests that the columns **Name** and **Quantity** do not exist in the result set. Here's an updated query that should work:

This query should work assuming the other tables exist in the database.

```
pyodbc.ProgrammingError) ('42S22', "[42S22] [Microsoft][ODBC Driver 17 for SQL Server][SQL Server]Invalid column name 'Name'. (207) [SQLExecDirectW]; [42S22] [Microsoft][ODBC Driver 17 for SQL Server][SQL Server]Invalid column name 'Name'. (207); [42S22] [Microsoft][ODBC Driver 17 for SQL Server][SQL Server]Invalid column name 'Name'. (207)"); [SQL: SELECT p.Name AS ProductName, YEAR(po.OrderDate) AS OrderYear, SUM(CASE WHEN DATEPART(QUARTER, po.OrderDate) = 1 THEN pod.OrderQty ELSE 0 END) AS Q1, SUM(CASE WHEN DATEPART(QUARTER, po.OrderDate) = 2 THEN pod.OrderQty ELSE 0 END) AS Q2, SUM(CASE WHEN DATEPART(QUARTER, po.OrderDate) = 3 THEN pod.OrderQty ELSE 0 END) AS Q3, SUM(CASE WHEN DATEPART(QUARTER, po.OrderDate) = 4 THEN pod.OrderQty ELSE 0 END) AS Q4, pm.Name AS ShipMethod FROM Production.Product p JOIN Purchasing.PurchaseOrderDetail pod ON p.ProductID = pod.ProductID JOIN Purchasing.PurchaseOrderHeader po ON pod.PurchaseOrderID = po.PurchaseOrderID JOIN Purchasing.ShipMethod pm ON po.ShipMethodID = pm.ShipMethodID GROUP BY p.Name, YEAR(po.OrderDate), pm.Name ] (Background on this error at: https://sqlalche.me/e/14/f405)
```

Cannot find the column "name" and therefore takes "ShipMethodID" as a reference.

I apologize for the error. It seems that the `Sales.ShipMethod` table is not present in the database. Here's an updated query that groups by `soh.ShipMethodID` instead:

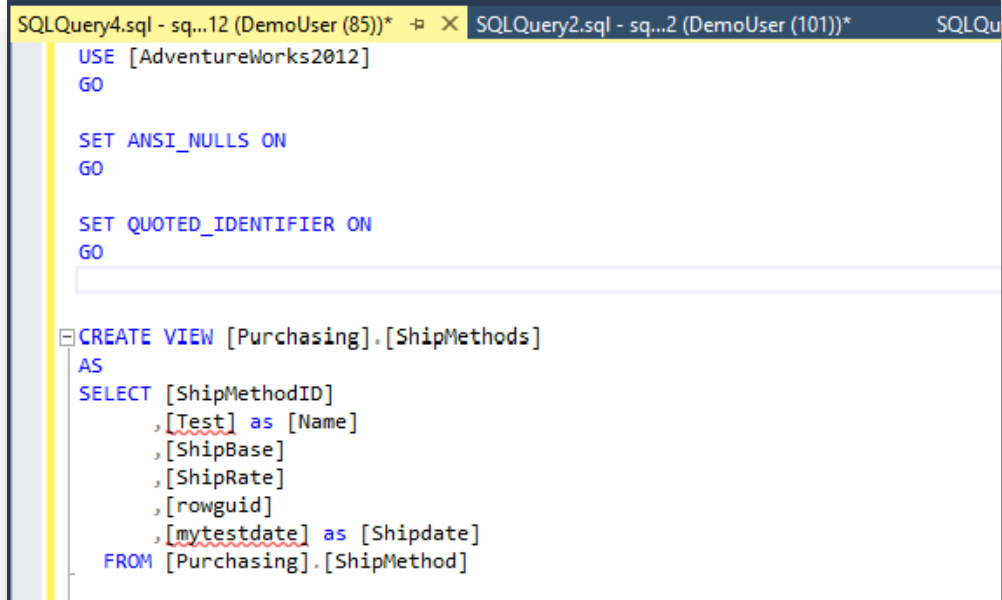
This should group the results by `soh.ShipMethodID` instead of `sm.Name`.

	ProductName	OrderYear	Q1	Q2	Q3	Q4	ShipMethodID
0	All-Purpose Bike Stand	2,013	0	6	65	65	1
1	All-Purpose Bike Stand	2,014	66	47	0	0	1
2	AWC Logo Cap	2,011	0	40	240	265	5
3	AWC Logo Cap	2,012	326	674	608	440	5
4	AWC Logo Cap	2,013	0	54	474	544	1
5	AWC Logo Cap	2,013	514	854	846	482	5
6	AWC Logo Cap	2,014	594	524	0	0	1
7	AWC Logo Cap	2,014	614	218	0	0	5
8	Bike Wash - Dissolver	2,013	0	22	209	247	1
9	Bike Wash - Dissolver	2,013	0	564	672	451	5

In SSMS, **create a View** via t-sql (adjust to your team DB):

```
USE [TEAMXX_LocalMasterDataDB]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

CREATE VIEW [Purchasing].[ShipMethods]
AS
SELECT [ShipMethodID] , [Test] as [Name]
, [ShipBase] , [ShipRate] , [rowguid]
, [mytestdate] as [Shipdate]
FROM [Purchasing].[ShipMethod]
```



The screenshot shows a SQL query window in SQL Server Enterprise Manager. The window title bar indicates it is 'SQLQuery4.sql - sq...12 (DemoUser (85))'. The query text is as follows:

```
USE [AdventureWorks2012]
GO

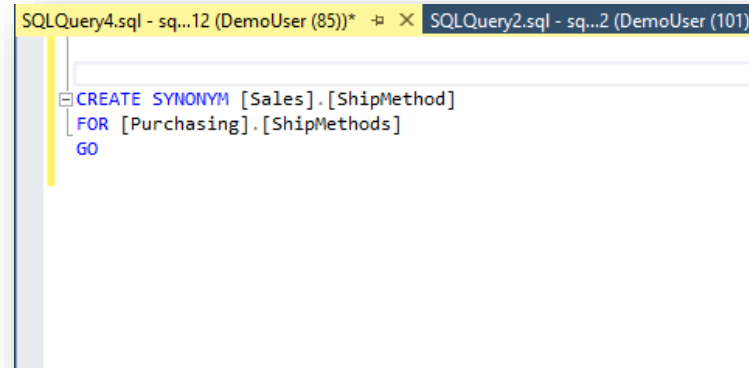
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE VIEW [Purchasing].[ShipMethods]
AS
SELECT [ShipMethodID]
, [Test] as [Name]
, [ShipBase]
, [ShipRate]
, [rowguid]
, [mytestdate] as [Shipdate]
FROM [Purchasing].[ShipMethod]
```

Create a synonym:

```
CREATE SYNONYM [Sales].[ShipMethod]
FOR [Purchasing].[ShipMethods]
GO
```

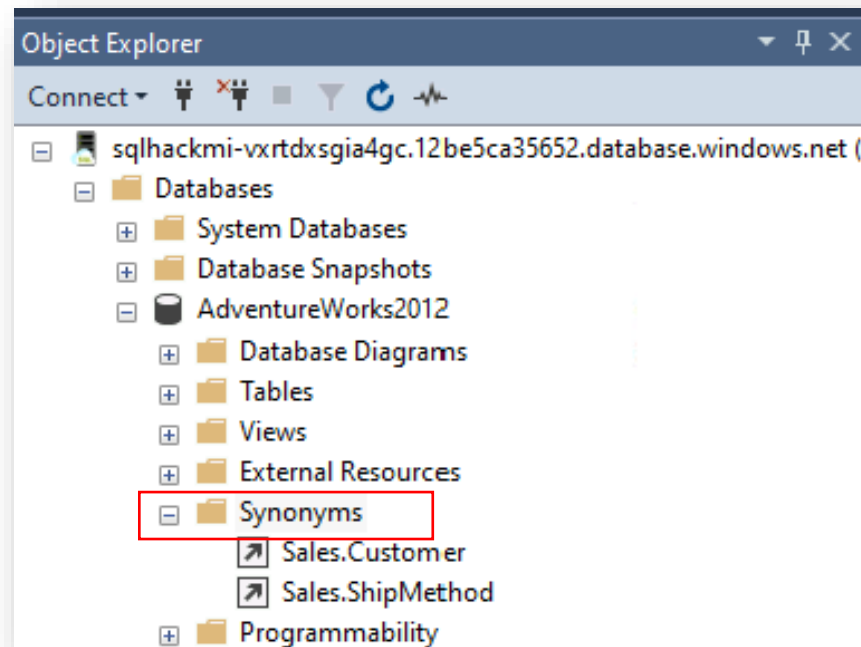


The screenshot shows a SQL Query Editor window with two tabs. The active tab is 'SQLQuery2.sql - sq...2 (DemoUser (101))'. The query text is:


```
CREATE SYNONYM [Sales].[ShipMethod]
FOR [Purchasing].[ShipMethods]
GO
```

 The query is highlighted in blue, indicating it is ready to be executed.

Review the synonym



Submit the question again and view the result

Find Quarterly Orders by Product. First column is Product Name, then year then four other columns, each for a quarter group by ship method

The screenshot shows the Natural Language Query interface. On the left, a sidebar contains options to 'Choose the app' (SQL Query Writing Assistant is selected), 'Settings', 'GPT Model' (ChatGPT), and 'FAQs'. A red box highlights the 'Ask me a question' section, which contains the text: 'Find Quarterly Orders by Product. First column is Product Name, then year then four other columns, each for a quarter group by ship method'. Below this is a 'Submit' button.

The main area displays the generated SQL code for the query:

```
SELECT
  p.Name AS ProductName,
  YEAR(soh.OrderDate) AS OrderYear,
  SUM(CASE WHEN DATEPART(QUARTER, soh.OrderDate) = 1 THEN sod.OrderQty ELSE 0 END) AS Q1,
  SUM(CASE WHEN DATEPART(QUARTER, soh.OrderDate) = 2 THEN sod.OrderQty ELSE 0 END) AS Q2,
  SUM(CASE WHEN DATEPART(QUARTER, soh.OrderDate) = 3 THEN sod.OrderQty ELSE 0 END) AS Q3,
  SUM(CASE WHEN DATEPART(QUARTER, soh.OrderDate) = 4 THEN sod.OrderQty ELSE 0 END) AS Q4,
  sm.Name AS ShipMethodName
FROM
  Sales.SalesOrderHeader AS soh
JOIN Sales.SalesOrderDetail AS sod ON soh.SalesOrderID = sod.SalesOrderID
JOIN Production.Product AS p ON sod.ProductID = p.ProductID
JOIN Sales.ShipMethod AS sm ON soh.ShipMethodID = sm.ShipMethodID
GROUP BY
  p.Name,
  YEAR(soh.OrderDate),
  sm.Name
ORDER BY
  p.Name,
  YEAR(soh.OrderDate),
  sm.Name
```

Below the SQL code is a table with the following data:

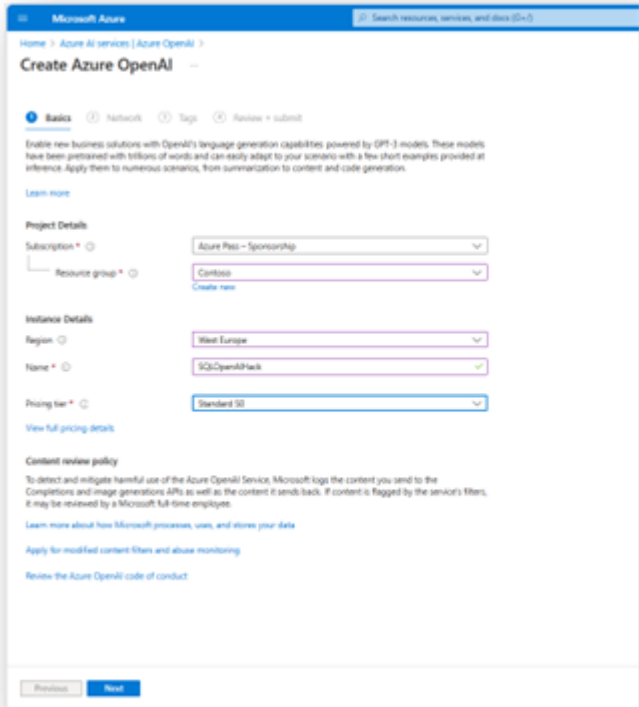
	ProductName	OrderYear	Q1	Q2	Q3	Q4	ShipMethodName
0	All-Purpose Bike Stand	2,013	0	6	65	65	XRQ - TRUCK GROUND
1	All-Purpose Bike Stand	2,014	66	47	0	0	XRQ - TRUCK GROUND
2	AWC Logo Cap	2,011	0	40	240	265	CARGO TRANSPORT 5
3	AWC Logo Cap	2,012	326	674	608	440	CARGO TRANSPORT 5
4	AWC Logo Cap	2,013	514	854	846	482	CARGO TRANSPORT 5
5	AWC Logo Cap	2,013	0	54	474	544	XRQ - TRUCK GROUND

Appendix

(OPTIONAL) Manual Deployment: You can follow the instructions below if you want to manually deploy the Azure OpenAI resource as well as the environment setup including Visual Studio Code, Git, Python, etc.

Azure OpenAI Service

Create an Azure OpenAI resource in an Azure subscription with a **GPT-35-Turbo** deployment and preferably a **GPT-4** deployment. Here we provide options for using both, but GPT-4 should be used to address more difficult & vague questions. We assume that your GPT-4 and ChatGPT deployments are in the same Azure Open AI resource.

Narrative	Screenshot	Notes
<p>See documentation for instructions:</p> <p>How-to: Create and deploy an Azure OpenAI Service resource - Azure OpenAI Microsoft Learn</p>		

Microsoft Azure

Home > Azure AI services > Azure OpenAI >

Create Azure OpenAI

Basics Network Tags Review + submit

Configure network security for your Azure AI services resource.

Type *

- ☒ All networks, including the internet, can access this resource.
- ☐ Selected networks, configure network security for your Azure AI services resource.
- ☐ Disabled, no networks can access this resource. You could configure private endpoint connections that will be the exclusive way to access this resource.

Previous Next

Microsoft Azure

Search resources, services, and docs (Ctrl+Q)

Home > Azure AI services > Azure OpenAI >

Create Azure OpenAI

Basics

Network

Tags

Review + submit

View automation template

TERMS

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. See the [Azure Marketplace Terms](#) for additional details.

Basics

Subscription

Resource group

Region

Name

Pricing tier

Azure Pass - Sponsorship

Contoso

West Europe

SQLOpenAIHack

Standard SD

Network

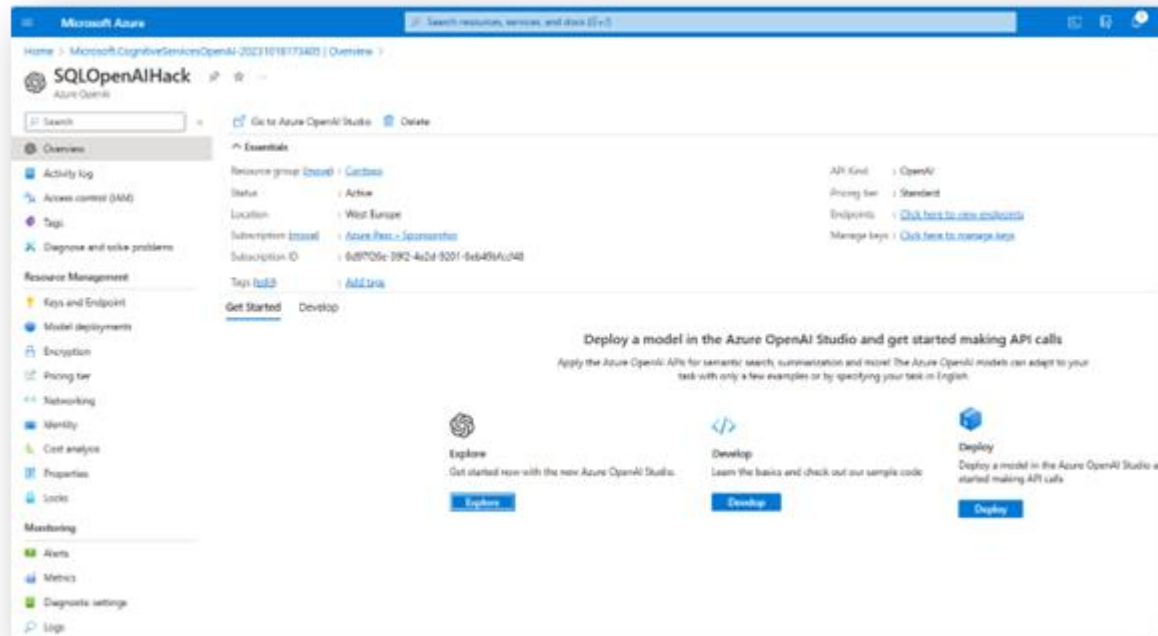
Type

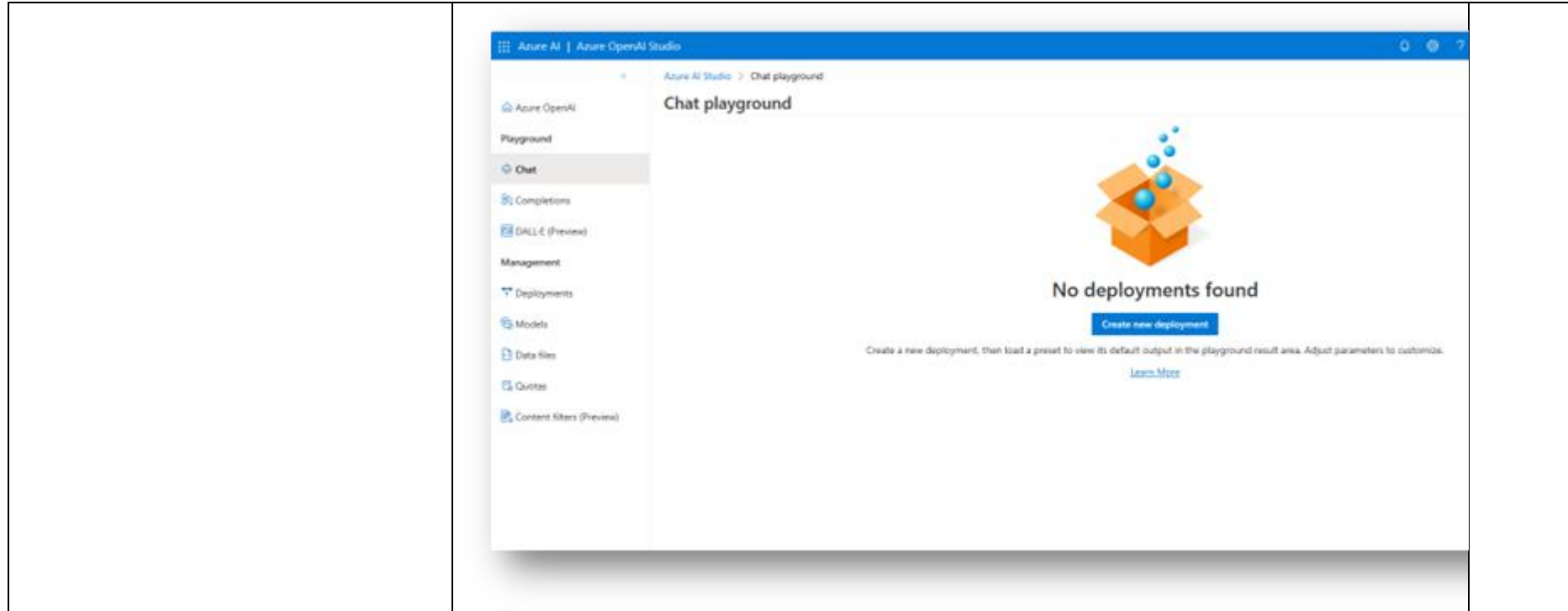
All networks, including the internet, can access this resource.

Previous

Next

Create





Deploy model

✕

Set up a deployment to make API calls against a provided base model or a custom model. Finished deployments are available for use. Your deployment status will move to succeeded when the deployment is complete and ready for use.

Select a model ⓘ

gpt-35-turbo ▾

Model version ⓘ

Auto-update to default ▾

Deployment name ⓘ

gpt-35-turbo ▾

⚙️ Advanced options ▾

Content Filter ⓘ

Default ▾

① 240K tokens per minute quota available for your deployment.

Tokens per Minute Rate Limit (thousands) ⓘ

120K

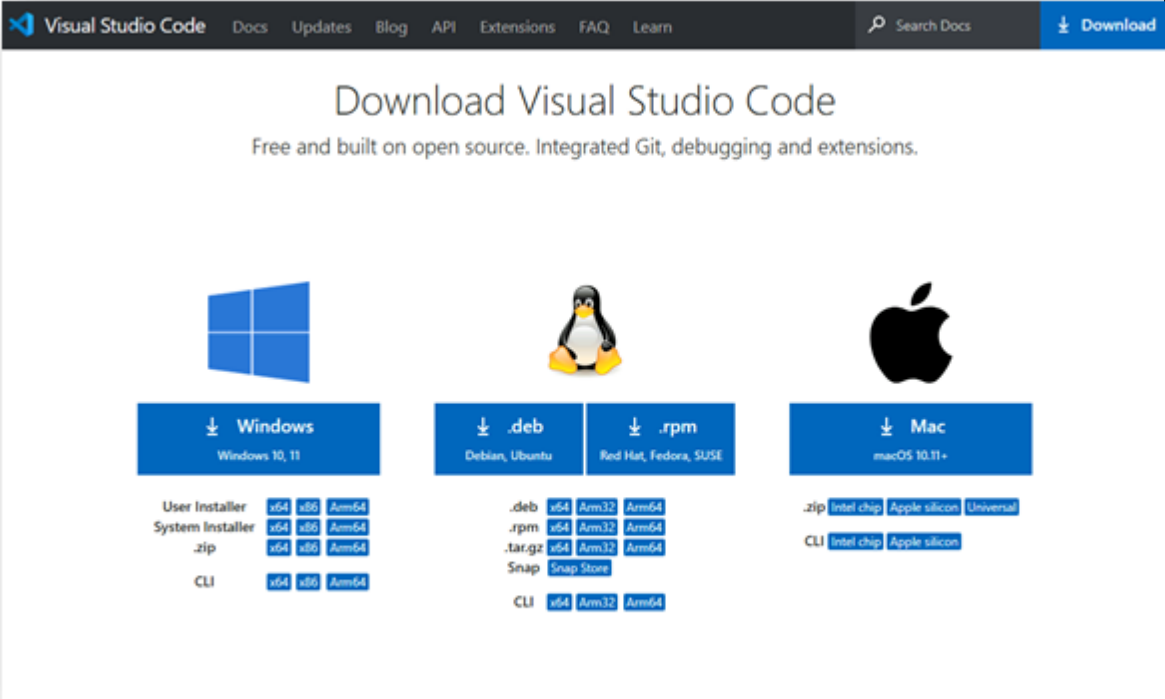
Corresponding requests per minute (RPM) = 720

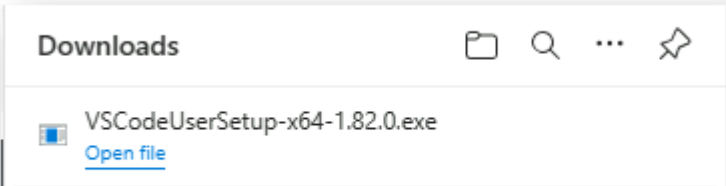
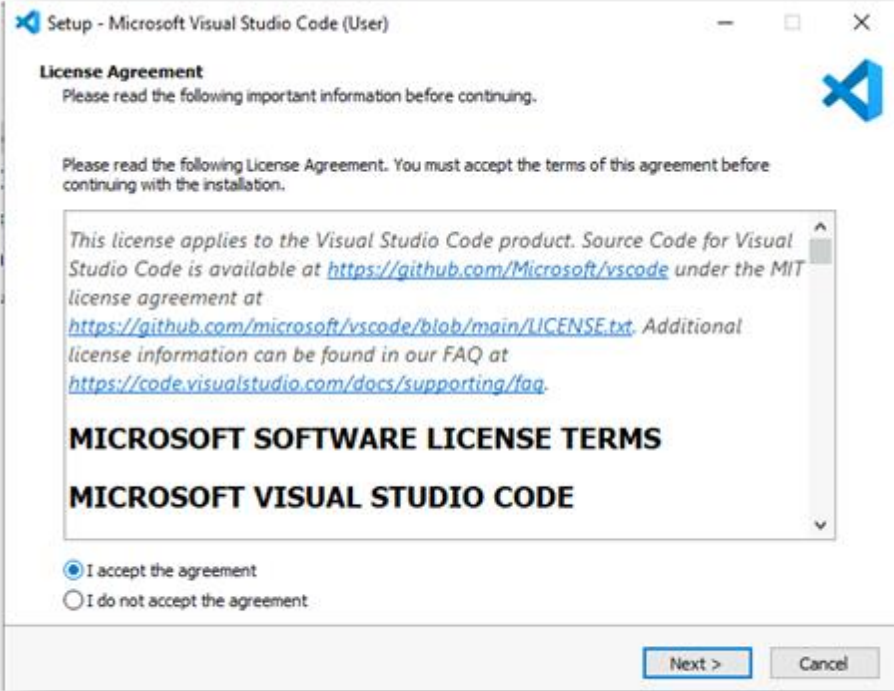
Create

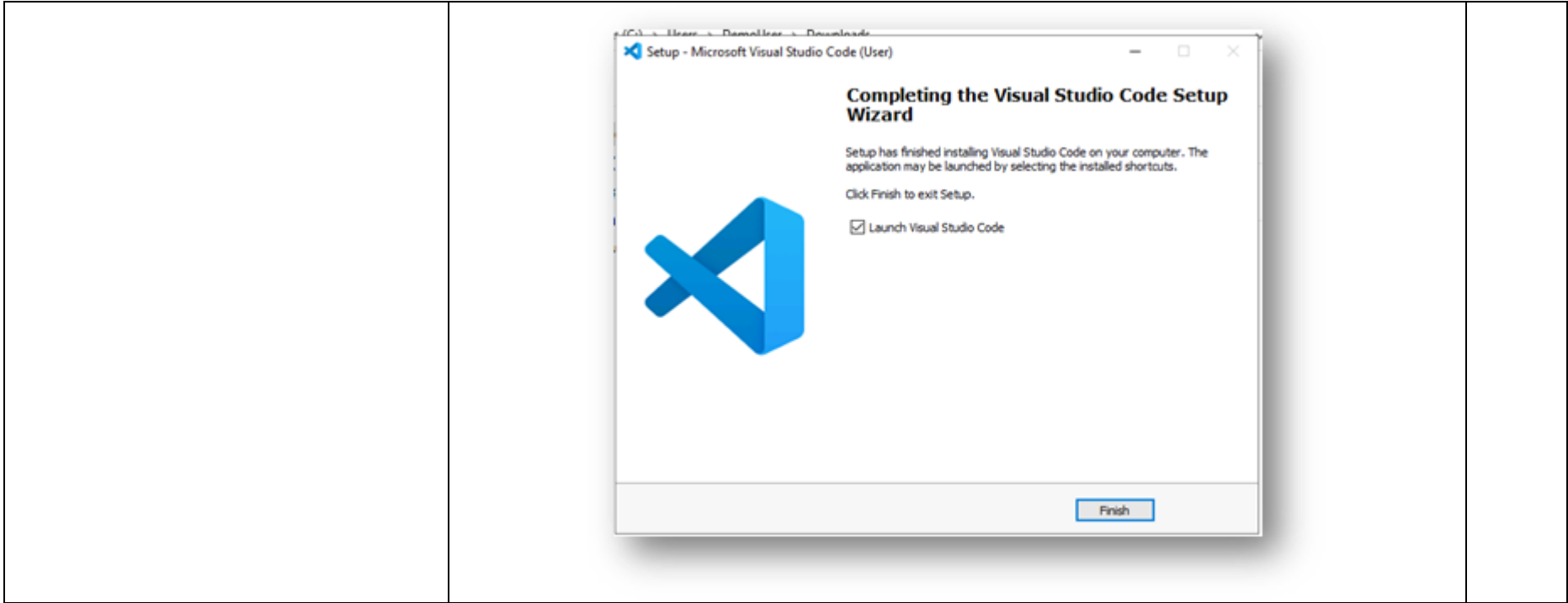
Cancel

Environment Setup

The environment has been provisioned for you upfront for this workshop. If you would like to manually set up the environment on your local machine, you can follow the instructions:

Narrative	Screenshot	Notes
<p>Download Visual Studio Code https://code.visualstudio.com/download</p>		

	 <p>A screenshot of a Windows File Explorer window titled 'Downloads'. It shows a single file named 'VSCodeUserSetup-x64-1.82.0.exe' with a blue icon. Below the filename is a link that says 'Open file'.</p>	
<p>Install Visual Studio Code</p>	 <p>A screenshot of the 'Setup - Microsoft Visual Studio Code (User)' window. The window displays the 'License Agreement' section. It asks the user to read the following important information before continuing. The text states: 'Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.' Below this is a scrollable text area containing the license terms. The text reads: 'This license applies to the Visual Studio Code product. Source Code for Visual Studio Code is available at https://github.com/Microsoft/vscode under the MIT license agreement at https://github.com/microsoft/vscode/blob/main/LICENSE.txt. Additional license information can be found in our FAQ at https://code.visualstudio.com/docs/supporting/faq.' Below the text area, there are two radio buttons: 'I accept the agreement' (which is selected) and 'I do not accept the agreement'. At the bottom right, there are 'Next >' and 'Cancel' buttons.</p>	



Download Git:

<https://git-scm.com/download/win>

Download for Windows

Click [here to download](#) the latest (2.42.0) 64-bit version of Git for Windows. This is the most recent [maintained build](#). It was released **12 days ago**, on 2023-08-30.

Other Git for Windows downloads

Standalone Installer

[32-bit Git for Windows Setup.](#)

[64-bit Git for Windows Setup.](#)

Portable ("thumbdrive edition")

[32-bit Git for Windows Portable.](#)

[64-bit Git for Windows Portable.](#)

Using winget tool

Install [winget](#) tool if you don't already have it, then type this command in command prompt or Powershell.

```
winget install --id Git.Git -e --source winget
```

The current source code release is version **2.42.0**. If you want the newer version, you can build it from [the source code](#).

Now What?

Now that you have downloaded Git, it's time to start using it.

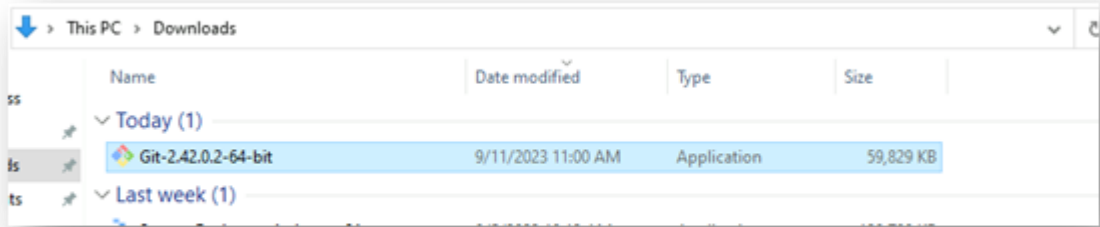

Downloads

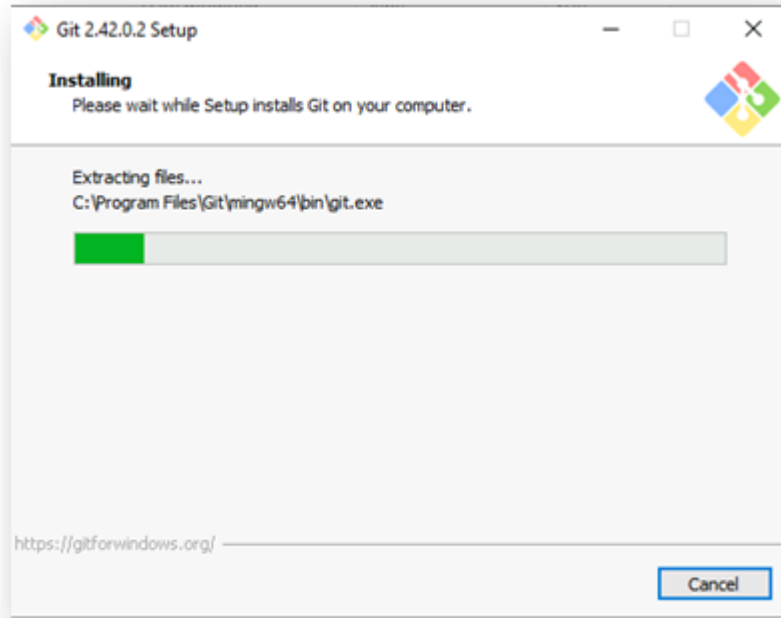
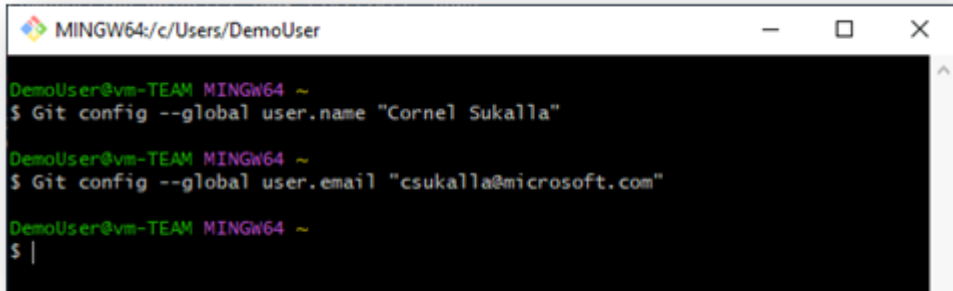


Git-2.42.0.2-64-bit.exe

0 B/s - 58.4 MB of 58.4 MB

[See more](#)

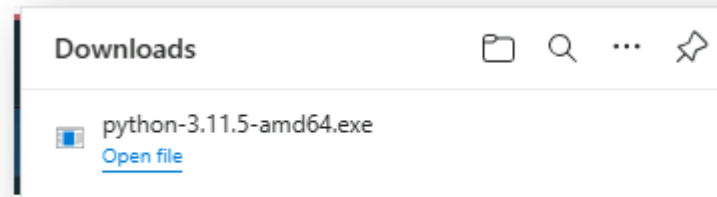
		
Install GIT		

	 <p>The image shows the 'Git 2.42.0.2 Setup' window. The title bar says 'Git 2.42.0.2 Setup'. Below the title bar, it says 'Installing' and 'Please wait while Setup installs Git on your computer.' There is a progress bar labeled 'Extracting files...' with the path 'C:\Program Files\Git\mingw64\bin\git.exe'. The progress bar is partially filled with green. At the bottom, there is a link 'https://gitforwindows.org/' and a 'Cancel' button.</p>	
(Optional) Set Git Variables	 <p>The image shows a terminal window titled 'MINGW64: c/Users/DemoUser'. The prompt is 'DemoUser@vm-TEAM MINGW64 ~'. The commands entered are: '\$ Git config --global user.name "Cornel Sukalla"' and '\$ Git config --global user.email "csukalla@microsoft.com"'. The prompt is now '\$ '.</p>	

Create a python environment with
version from 3.7 and 3.10

Download Python:

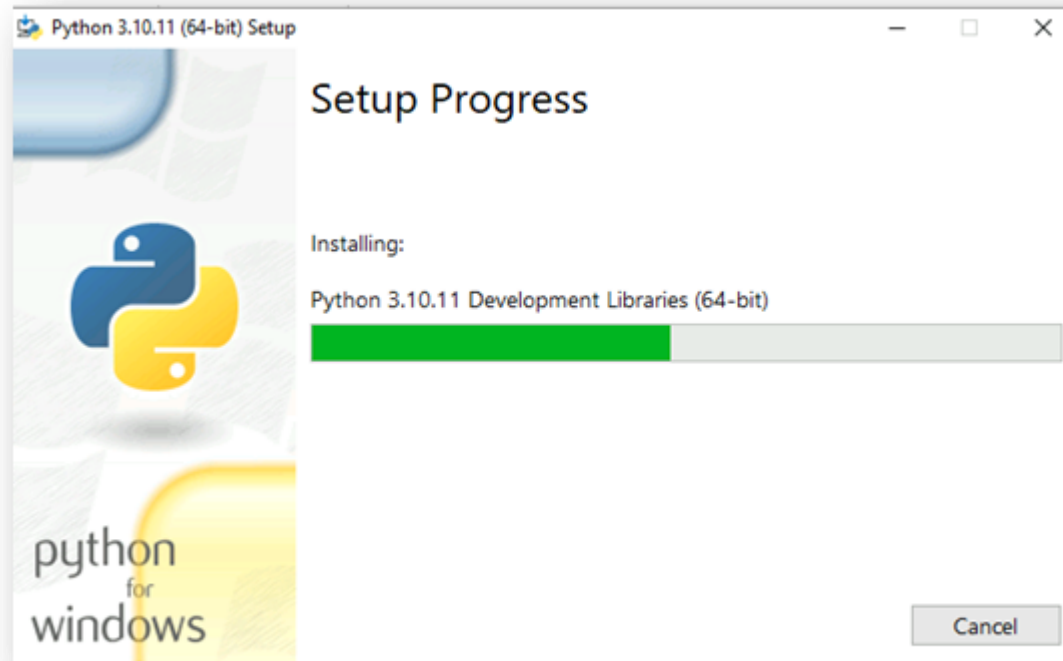
<https://www.python.org/downloads/>

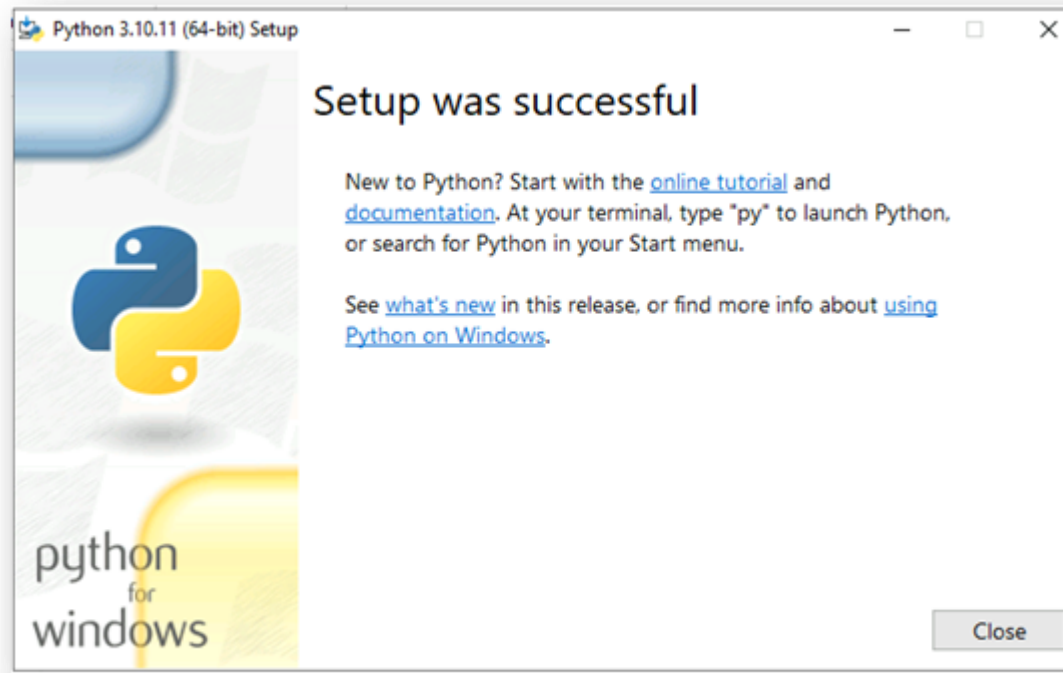


Installation Python

Important:

- Python and the pip package manager must be in the path in Windows for the setup scripts to work.
- Ensure you can run `python --version` from console. On Ubuntu, you might need to run `sudo apt install python-is-python3` to link `python` to `python3`.





```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/powershell

PS C:\Users\DemoUser> python --version
Python 3.10.11
PS C:\Users\DemoUser>
```