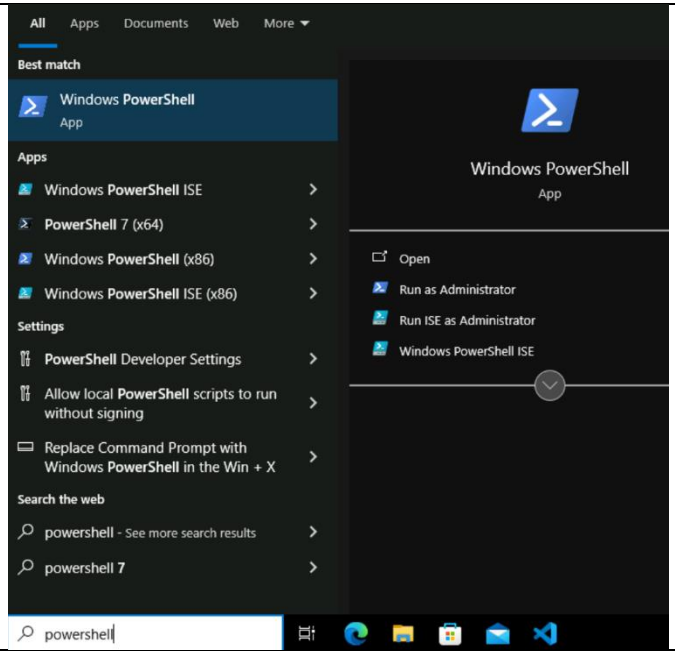
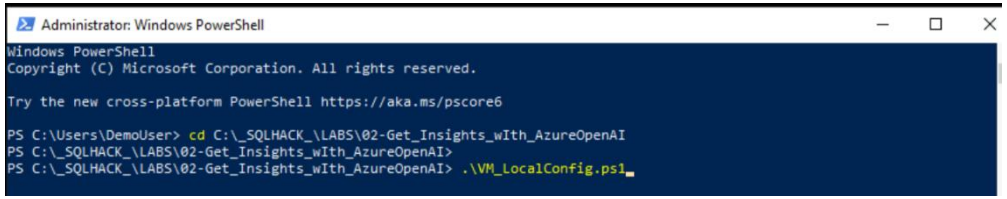
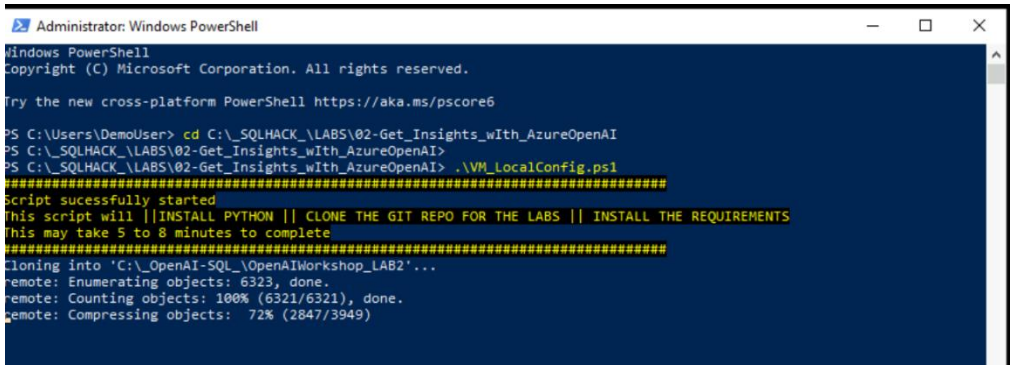
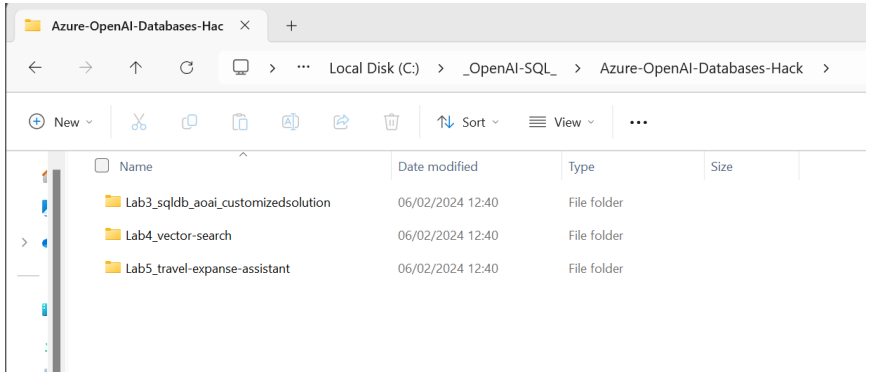
Lab 3 - SQL&GPT: Deploy Application and test SQL Writing & Data Analysis Assistants

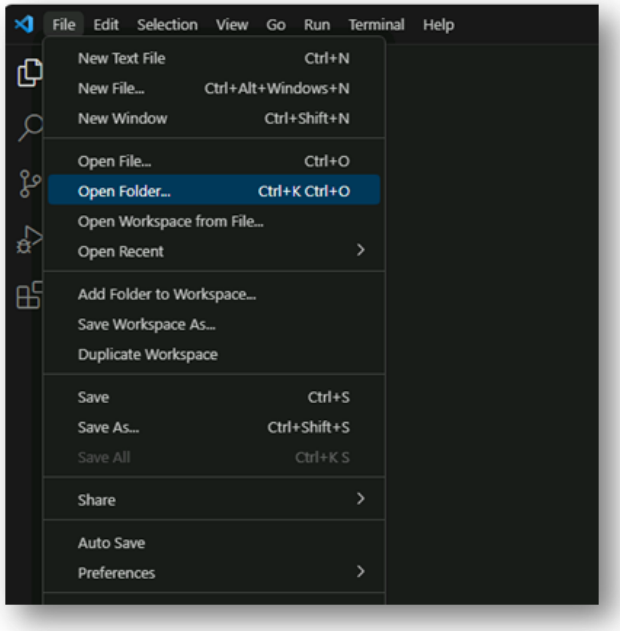
Contents

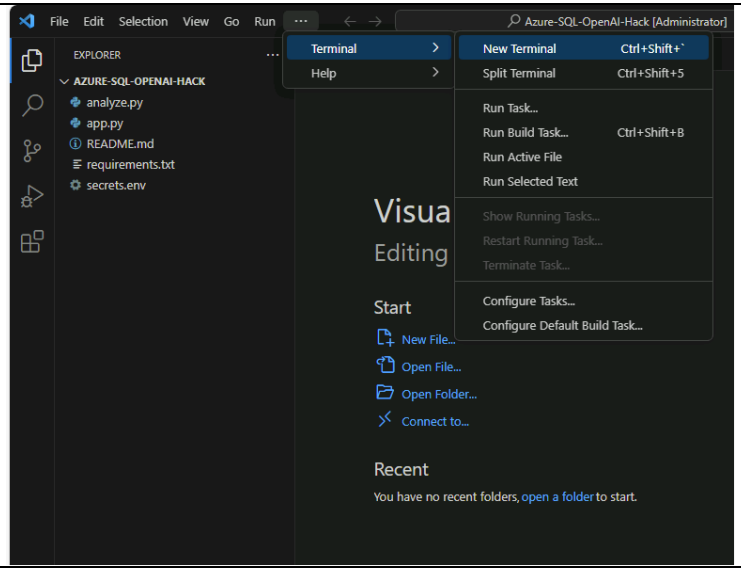
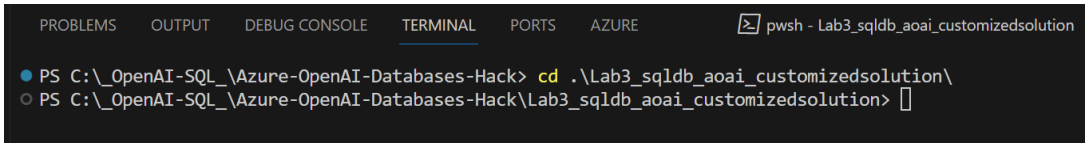
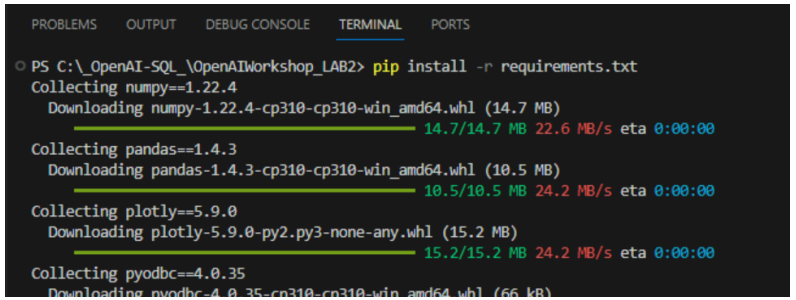
1. Install the application locally

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**
Switch back to your Team VM. 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).		
In PowerShell command line, **navigate to following folder** with the command: ***cd C:_SQLHACK_\LABS\02-Get_Insights_wIth_AzureOpenAI***		

Run the file ***VM_LocalConfig.ps1*** by typing in: ***.\VM_LocalConfig.ps1*** Hit enter and let the script run. This might take some time.		This script will: - Install Python - clone the git Repo for the Labs 2 & 3 - Install the Python Requirements - Set some environment variables
The repository with the 3 labs was cloned for you. Check by navigating to path in the file explorer: ***C:_OpenAI-SQL_\Azure-OpenAI-Databases-Hack*** This is the folder/project you will open in Visual Studio Code.		

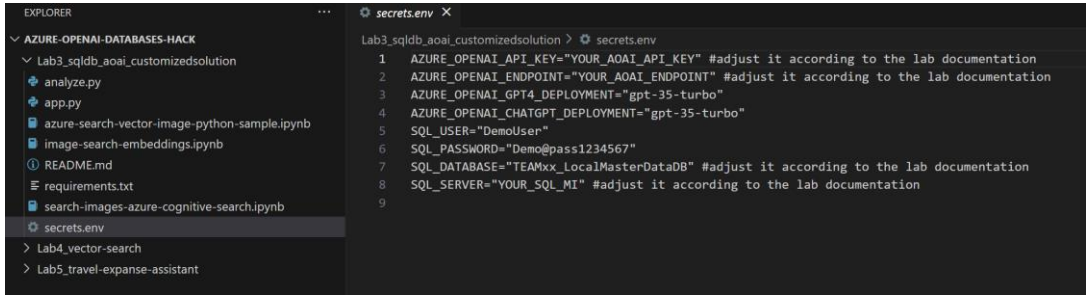
Open **Visual Studio Code** in the Windows start menu. Import the project/folder via ***File > Open Folder.*** Then select the *folder "**C:_OpenAI-SQL_\Azure-OpenAI-Databases-Hack**"* When prompted, check the "Trust the authors of all files" option and click **Yes, I trust the authors**.		

Open a **terminal** via ***Terminal Tab > New Terminal***		
Navigate to the lab 3 subfolder with the cd command: ***cd .\Lab3_sqldb_aoai_customizedsolution***		
Check if the needed **requirements** are already installed with the command: ***pip install -r requirements.txt***		

Now set the **credential values** within the application code (environment file).

Navigate to the ***secrets.env*** file with the secrets in Lab3 folder.

These should be replaced with the values of the services we use from the Hack subscriptions.

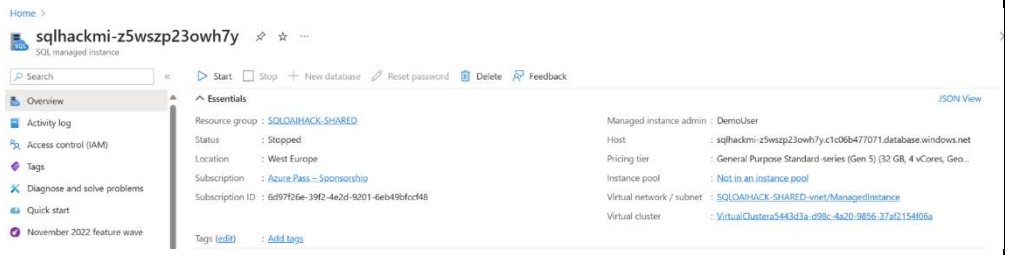


Update all your credentials according to the ones on the right side. Make sure you fill in the **highlighted values**.
You can **copy the credentials** on the right side to your env file.

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"

For the **Azure OpenAI Endpoint + API Key**, use the values according to your team number.

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

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

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.

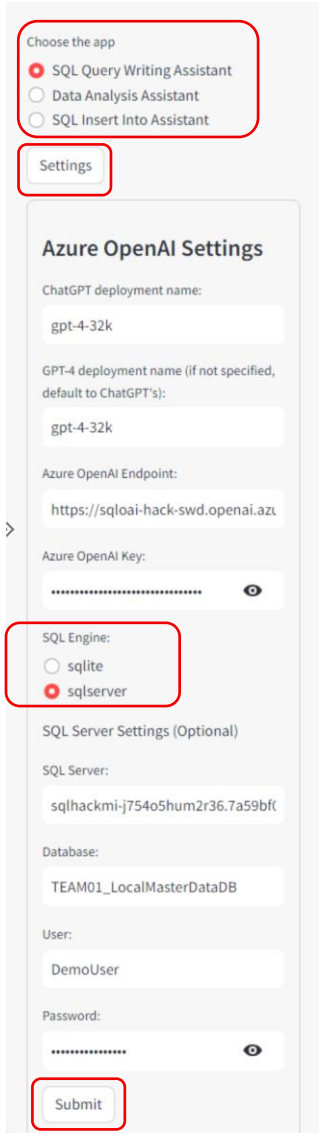
On the left side, select the assistant app "**SQL Query writing Assistant**".

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.

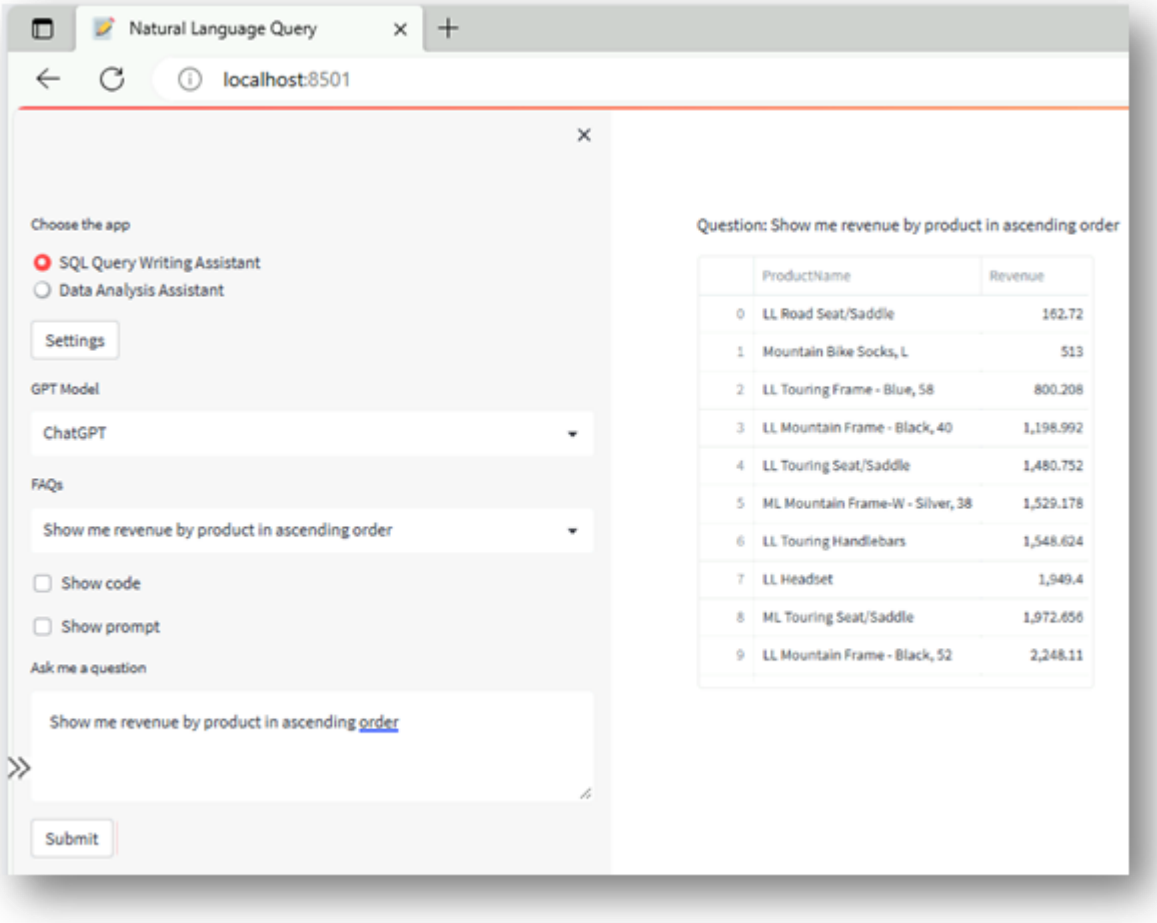
Side note:
The option "sqlite" is a built-in server by Python for you to test out.
In this workshop, we will focus on using our own managed instance (option "**sql server**").

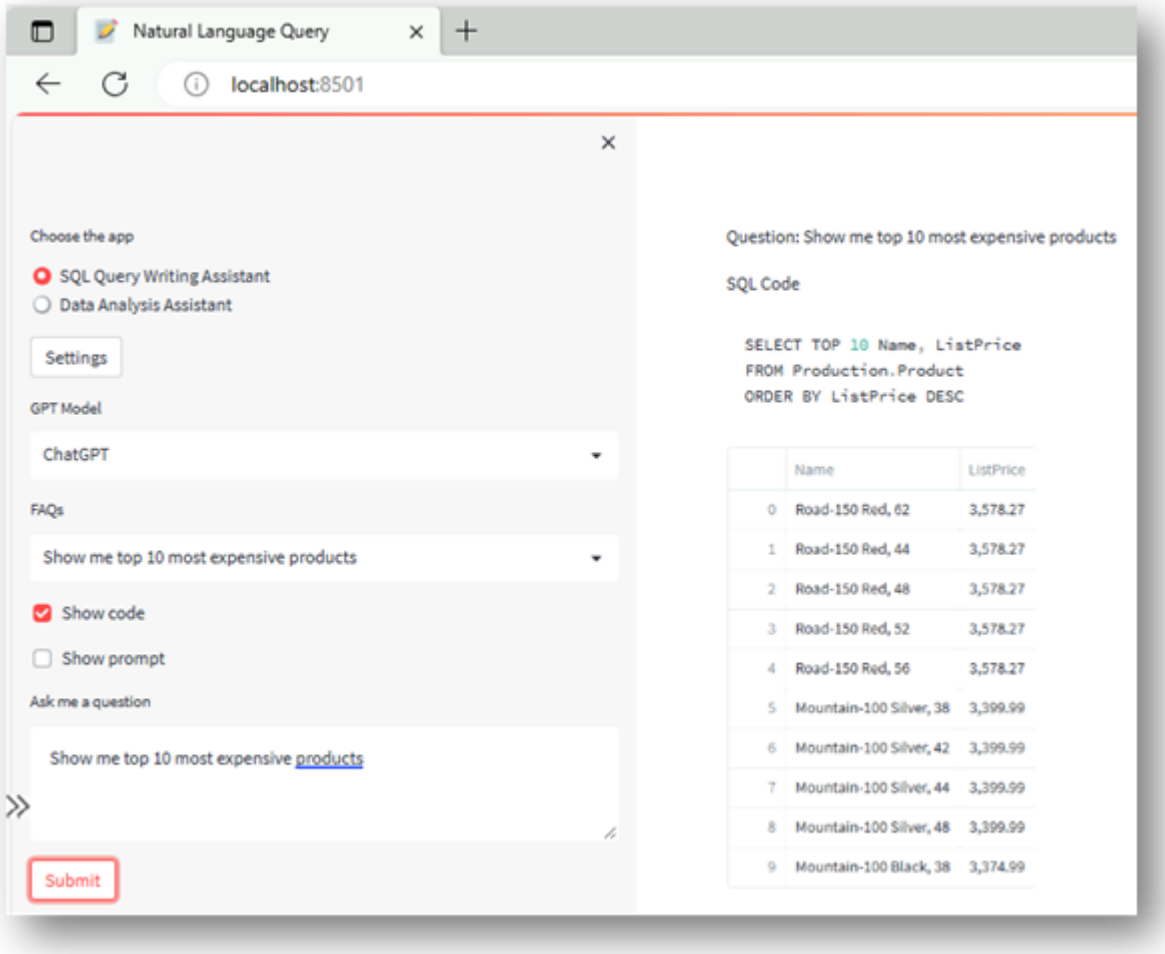


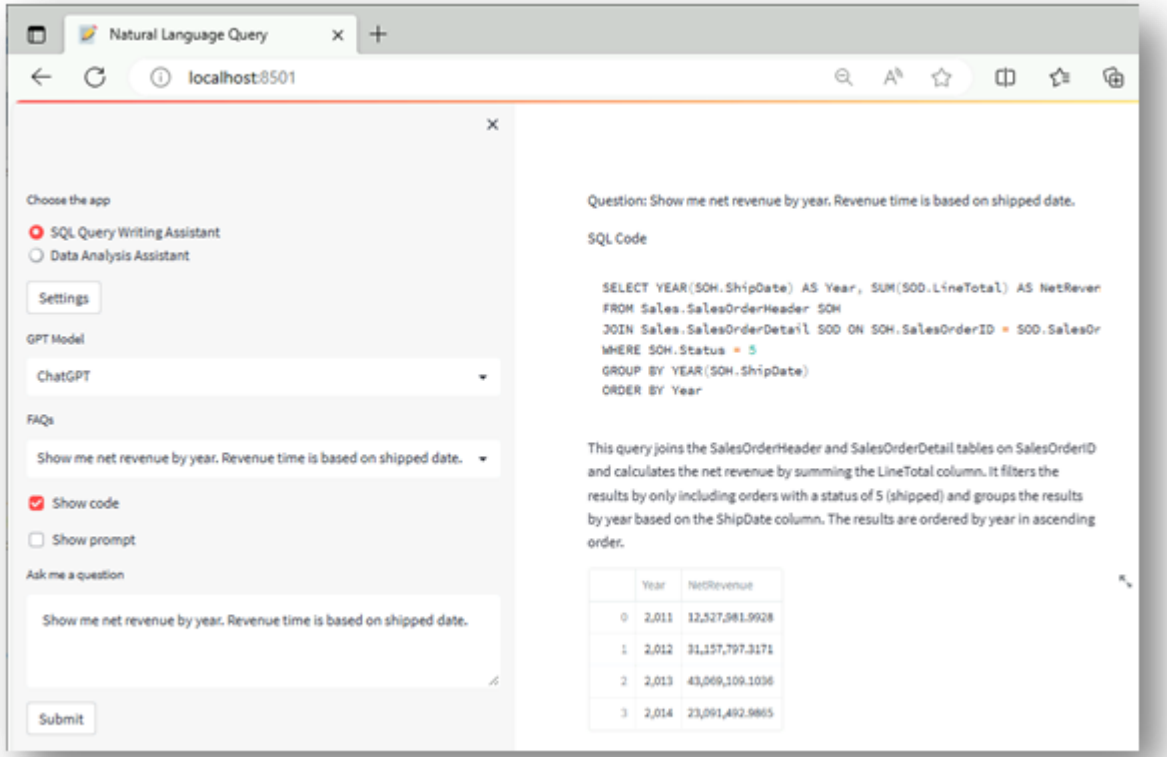
Let's test the **SQL Query writing Assistant**. Make sure you have selected this assistant.

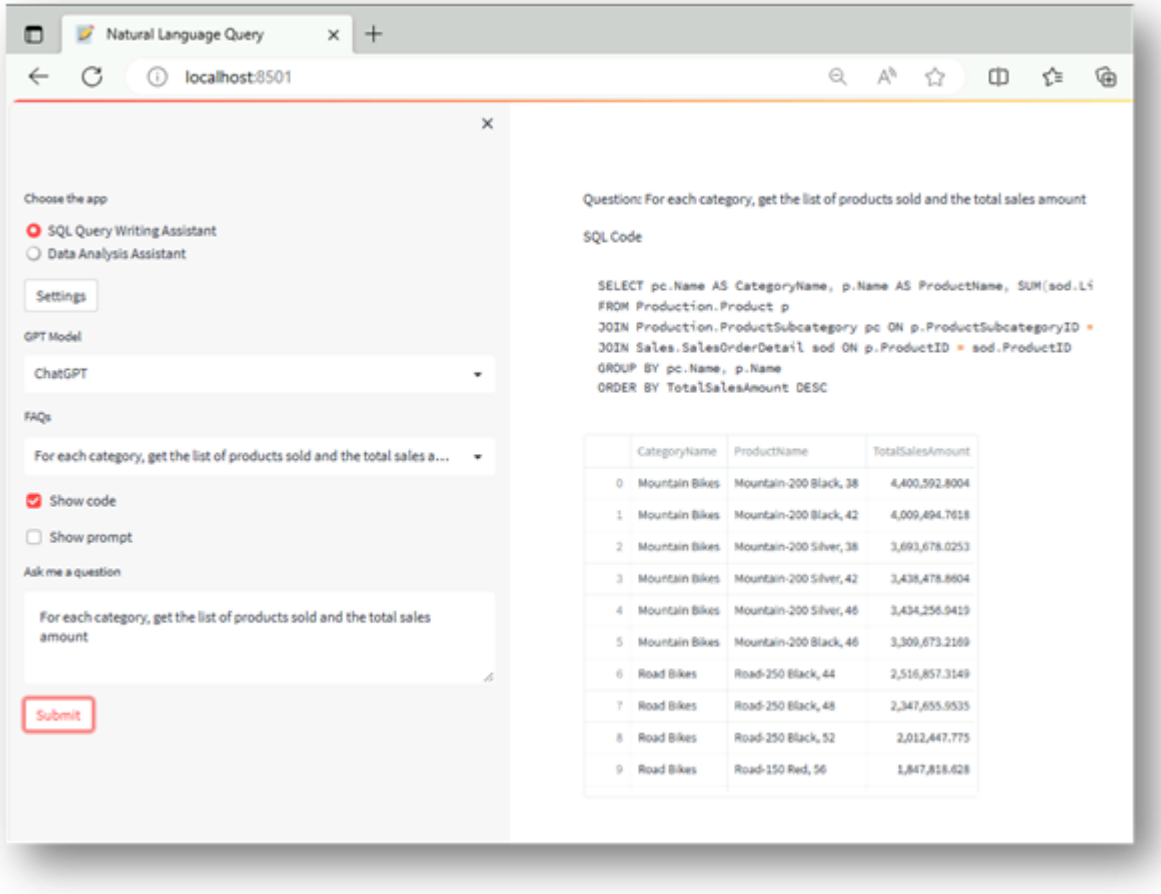
Under "**Ask me a question**" below, you can try out your first sample queries in natural language.
Past the sample 1 below and hit the **Submit** button after.

Sample 1: Show me revenue by product in ascending order

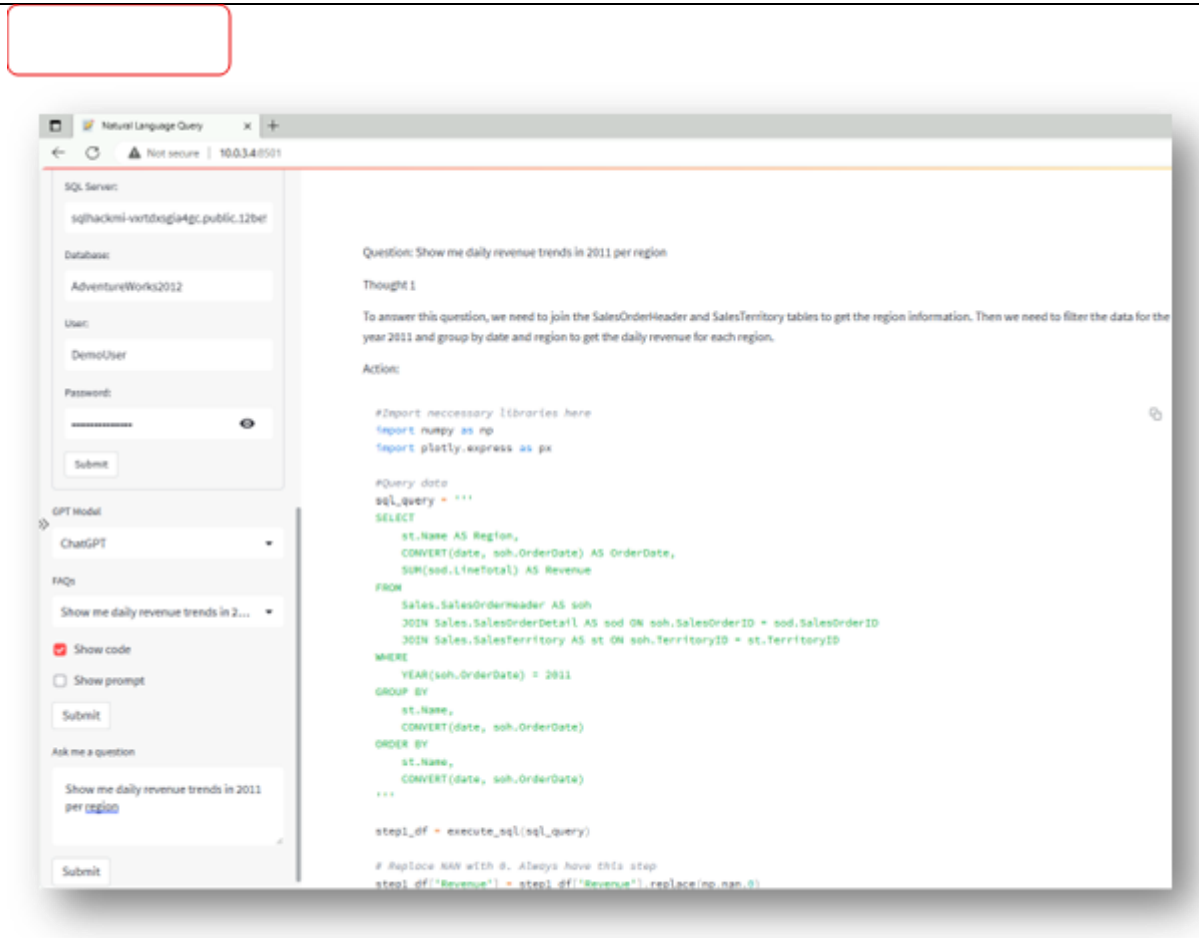


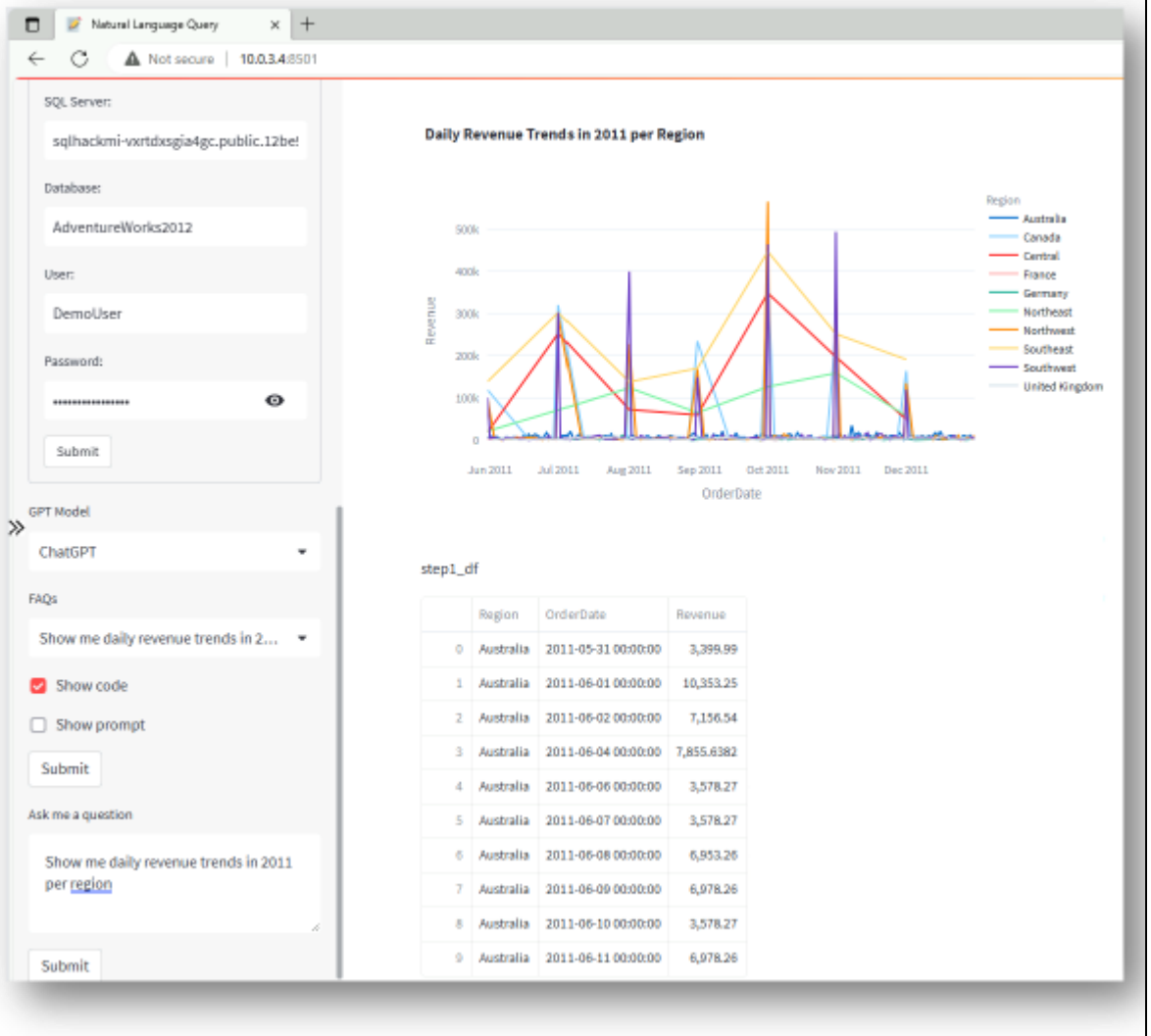
***Sample 2:* Show me top 10 most expensive products**		

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

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

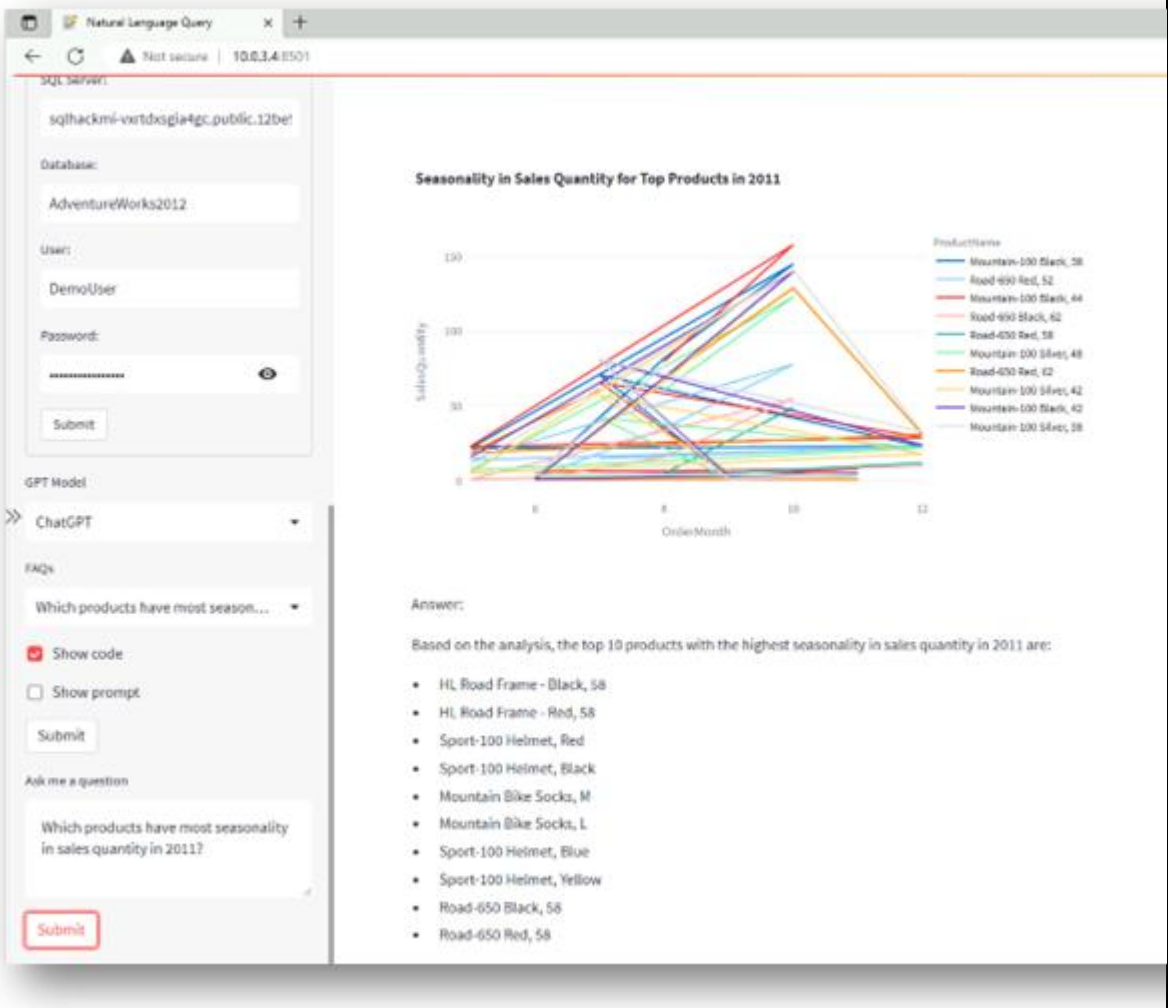
Now, let's test the second assistant "Data Analysis Assistant" to generate further insights – also using natural language.

Narrative	Screenshot	Notes
In the UI, switch to the "**Data Analysis Assistant**". Make sure to select "**sqlserver**" as SQL Engine under connecting settings. Then hit **submit**. Now, you can paste the following query and hit submit: ***Sample 1: Show me daily revenue trends in 2011 per region***		


Natural Language Query
Not secure | 10.0.3.4:8501
SQL Server:
sqlhackmi-vxrtdxsgja4gc.public.12be!
Database:
AdventureWorks2012
User:
DemoUser
Password:
Submit
GPT Model
ChatGPT
FAQs
Show me daily revenue trends in 2...
Show code
Show prompt
Submit
Ask me a question
Show me daily revenue trends in 2011 per region
Submit
Daily Revenue Trends in 2011 per Region
Region
Australia
Canada
Central
France
Germany
Northeast
Northwest
Southeast
Southwest
United Kingdom
Revenue
500k
400k
300k
200k
100k
0
Jun 2011
Jul 2011
Aug 2011
Sep 2011
Oct 2011
Nov 2011
Dec 2011
OrderDate
step1_df
Region
OrderDate
Revenue
0 Australia 2011-05-31 00:00:00 3,399.99
1 Australia 2011-06-01 00:00:00 10,353.25
2 Australia 2011-06-02 00:00:00 7,156.54
3 Australia 2011-06-04 00:00:00 7,855.6382
4 Australia 2011-06-06 00:00:00 3,578.27
5 Australia 2011-06-07 00:00:00 3,578.27
6 Australia 2011-06-08 00:00:00 6,953.26
7 Australia 2011-06-09 00:00:00 6,978.26
8 Australia 2011-06-10 00:00:00 3,578.27
9 Australia 2011-06-11 00:00:00 6,978.26

Sample 2:

Which products have most seasonality in sales quantity in 2011?



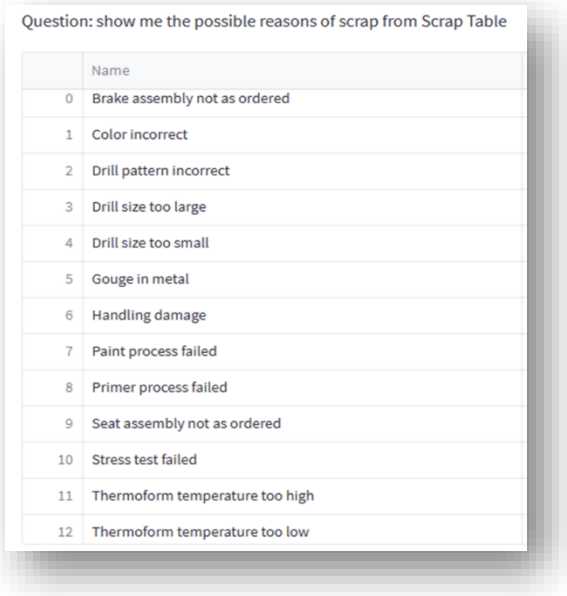
3. Insert new data via natural language

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).

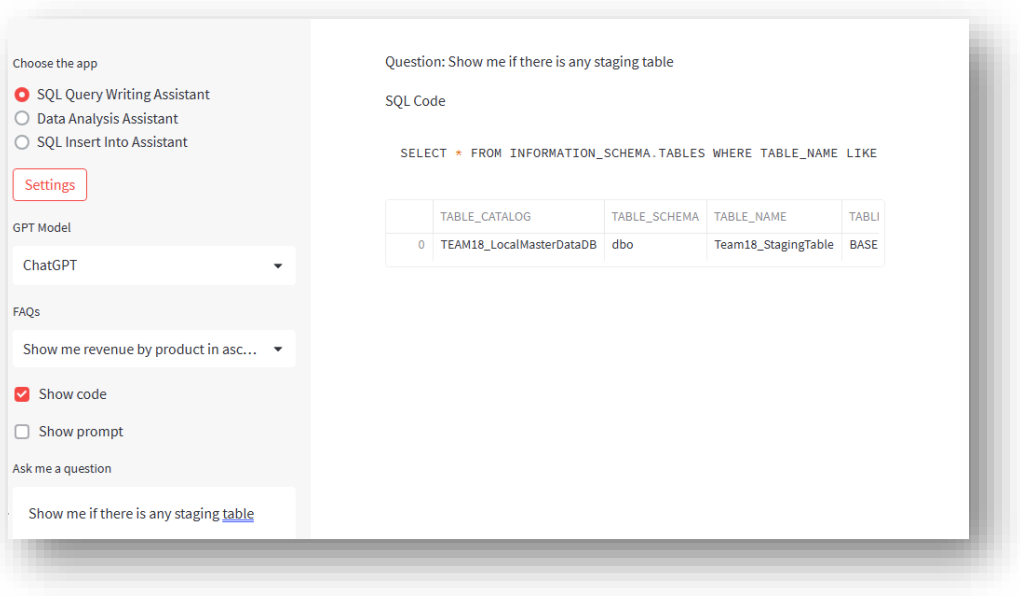
Scenario: In the LocalMasterDataDB manufacturing plant, the Scrap Redemption Center plays a crucial role in documenting and managing scrap data. To provide information about the reason of part rejection, the worker enters information about the type of scrap (e.g., "Gouge in metal," " Drill size too large").

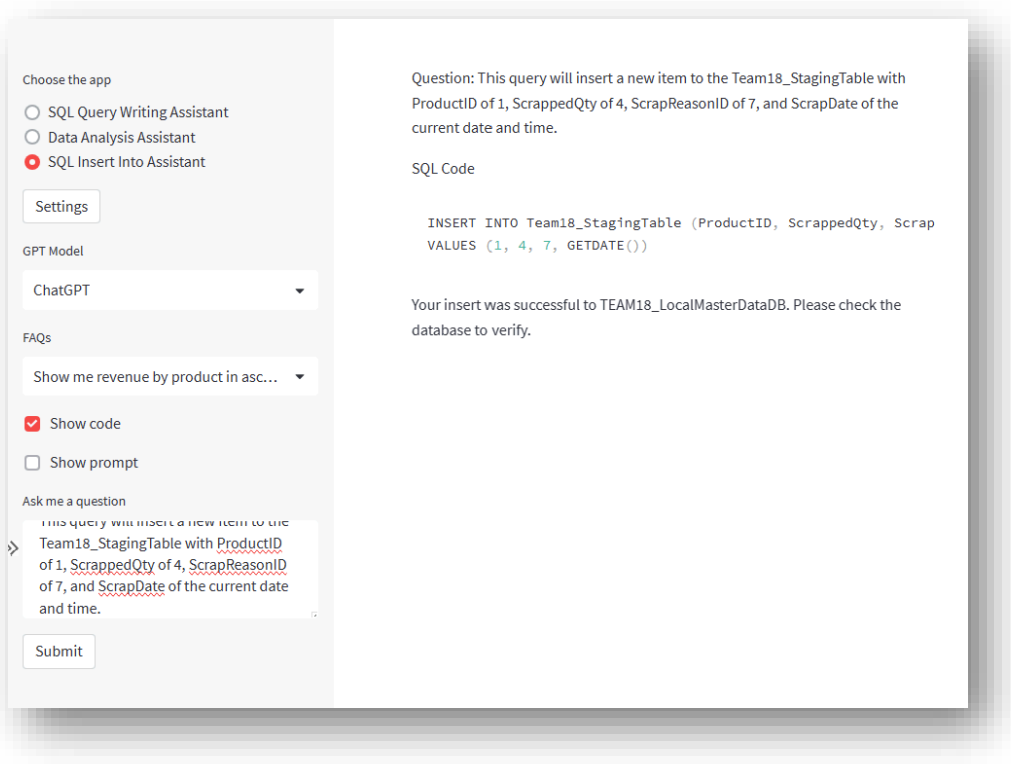
Narrative	Screenshot	Notes
Select the **SQL Query Writing Assistant.** Make sure in settings, **sqlserver** is selected as SQL Engine. Click **Submit** to save the settings. Check the reasons for the scrap parts with the following prompt: ***Show me the possible reasons of Scrap Parts***		

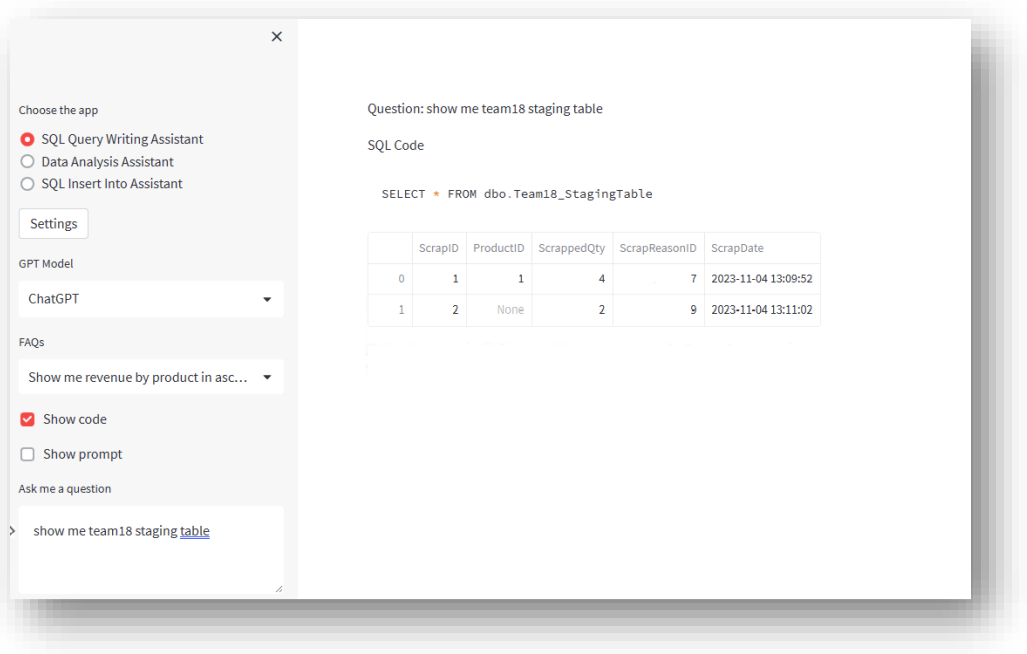
Insert with the Staging Table

Narrative	Screenshot/Code	Notes

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**".
 	
Make sure you are in the right context (your Team DB) before executing the query.
 	
Create a **staging table** with the T-SQL statement on the right and insert your TeamXX name for the table.
	`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)`
	`);`	

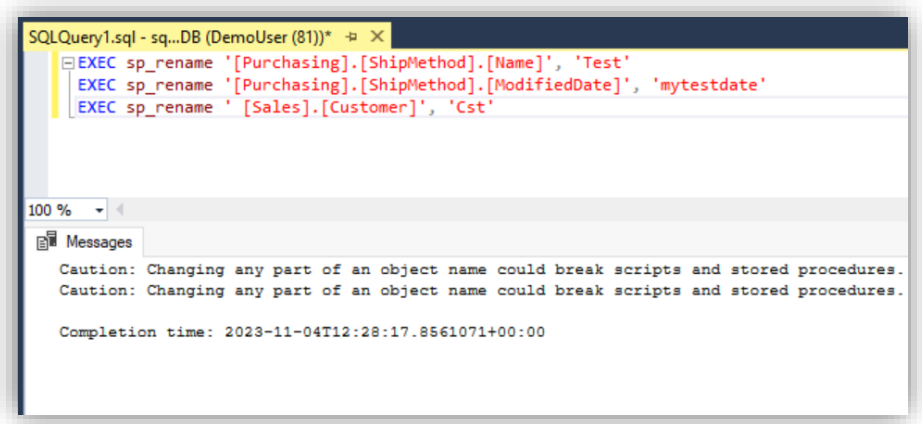
Check if the table is already there with the help of **SQL Query Writing Assistant**. Use the prompt: ***Show me if there is any staging table***		

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.***		

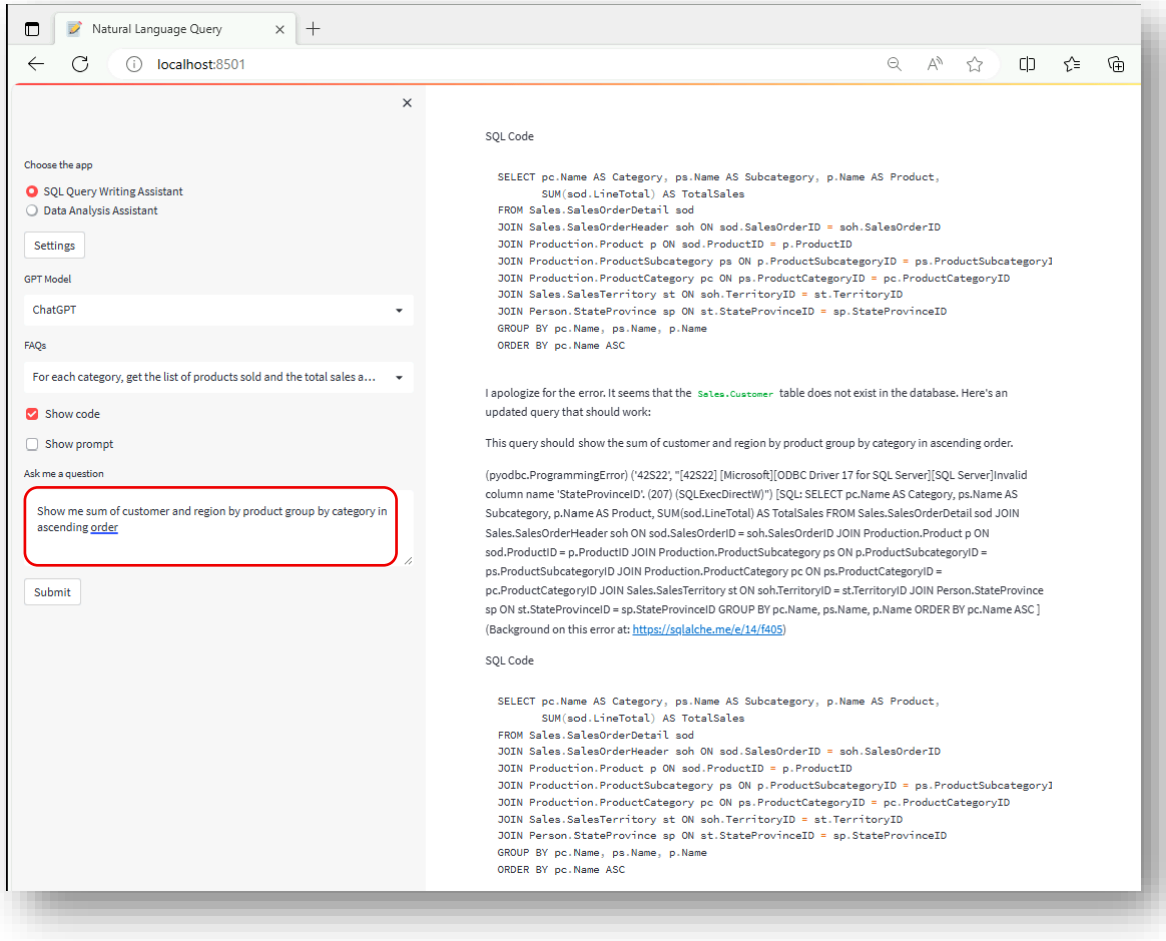
Insert a new value and check that the primary key (ScrapID) is automatically increased with the following prompt (again, adjust to your staging table name: ***Insert a new item to the TeamXX_StagingTable with ScrappedQty of 2, ScrapReasonID of 9, and ScrapDate of the current date and time.*** **Verify** in **SSMS** or in the **SQL Server Writing Assistant** that the item was inserted and there was an automatic increase of ScrapID.		

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.

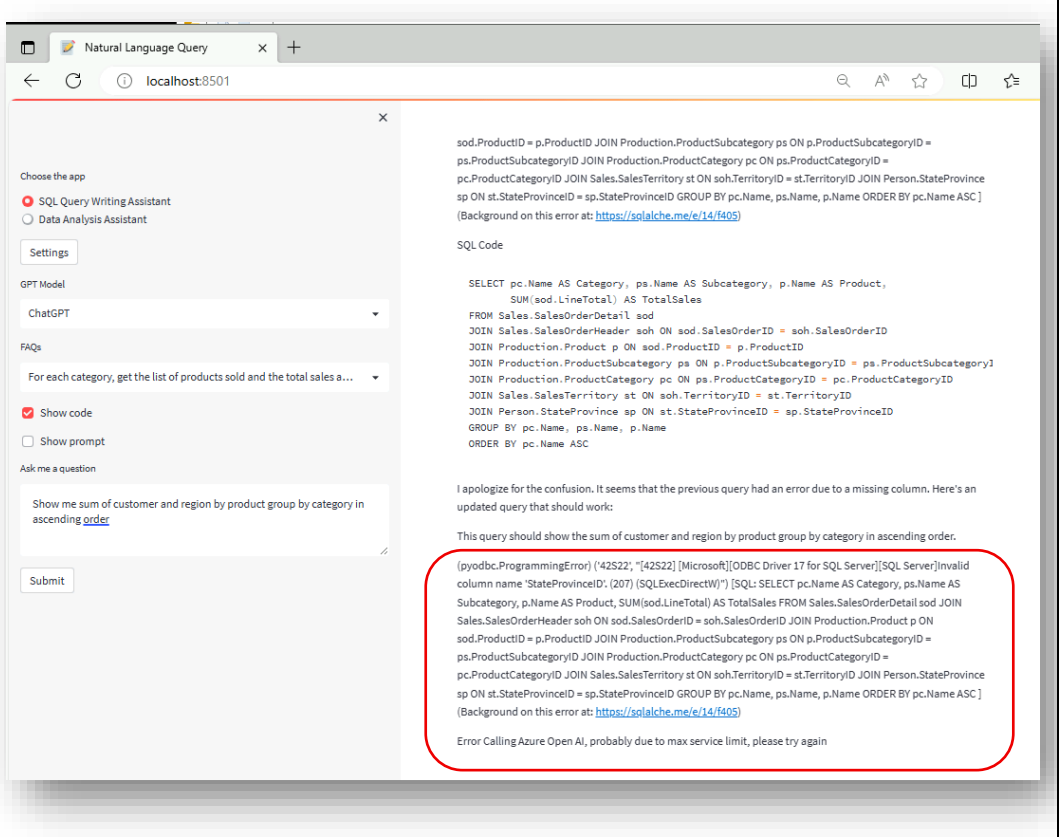
Narrative	Screenshot	Notes
First, let's intentionally create "quality issues within our DB. In SSMS, open a query window on your TeamXX_LocalMasterDataDB and execute the following T-SQL statement: *EXEC sp_rename '[Purchasing].[ShipMethod].[Name]', 'Test'* *EXEC sp_rename '[Purchasing].[ShipMethod].[ModifiedDate]',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

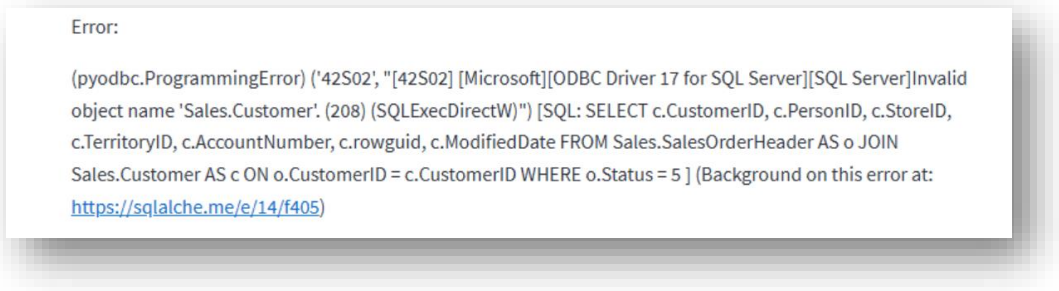
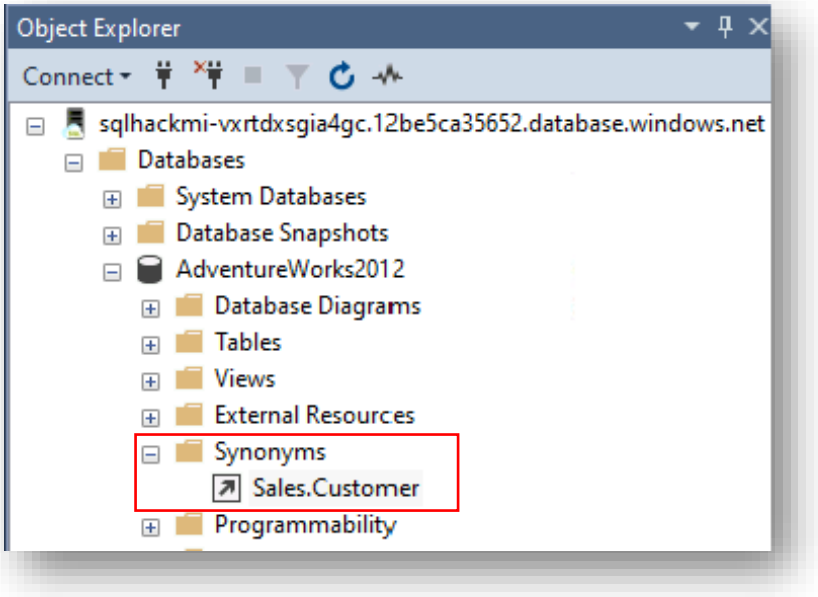
Now, let's try some samples which will run into errors to understand potential data quality issues we would have to resolve.

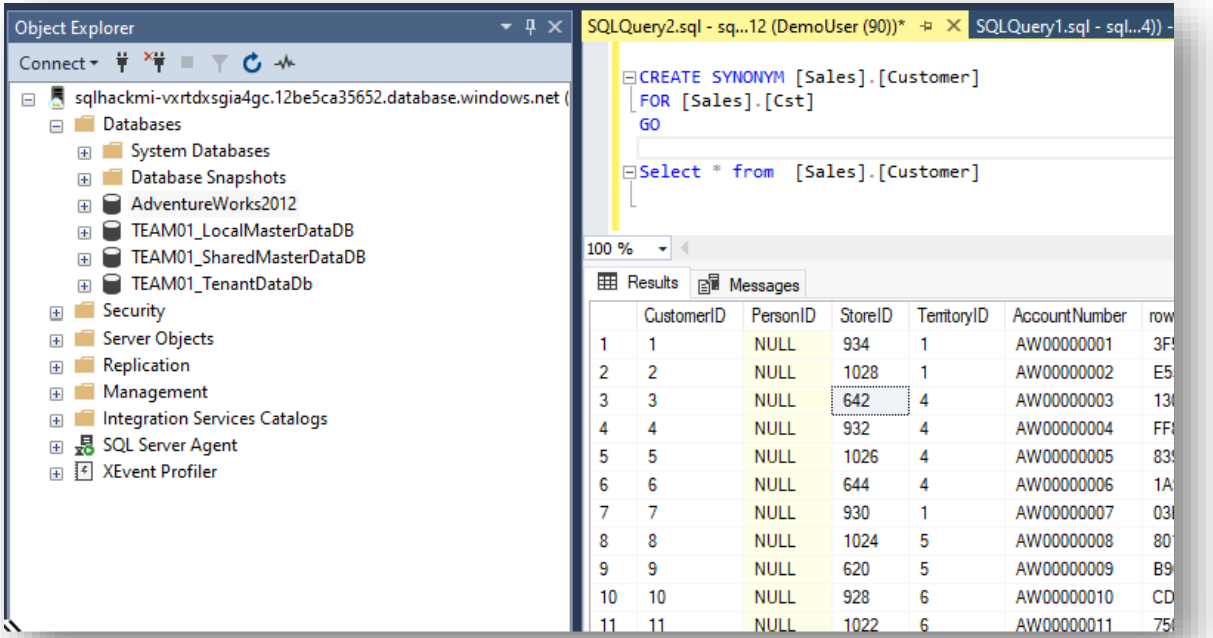
Narrative	Screenshot	Notes
Sample 1: Copy the text in the question box, still using SQL Query Assistant: ***Show me customer by product group by category in ascending order***		

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'.*".

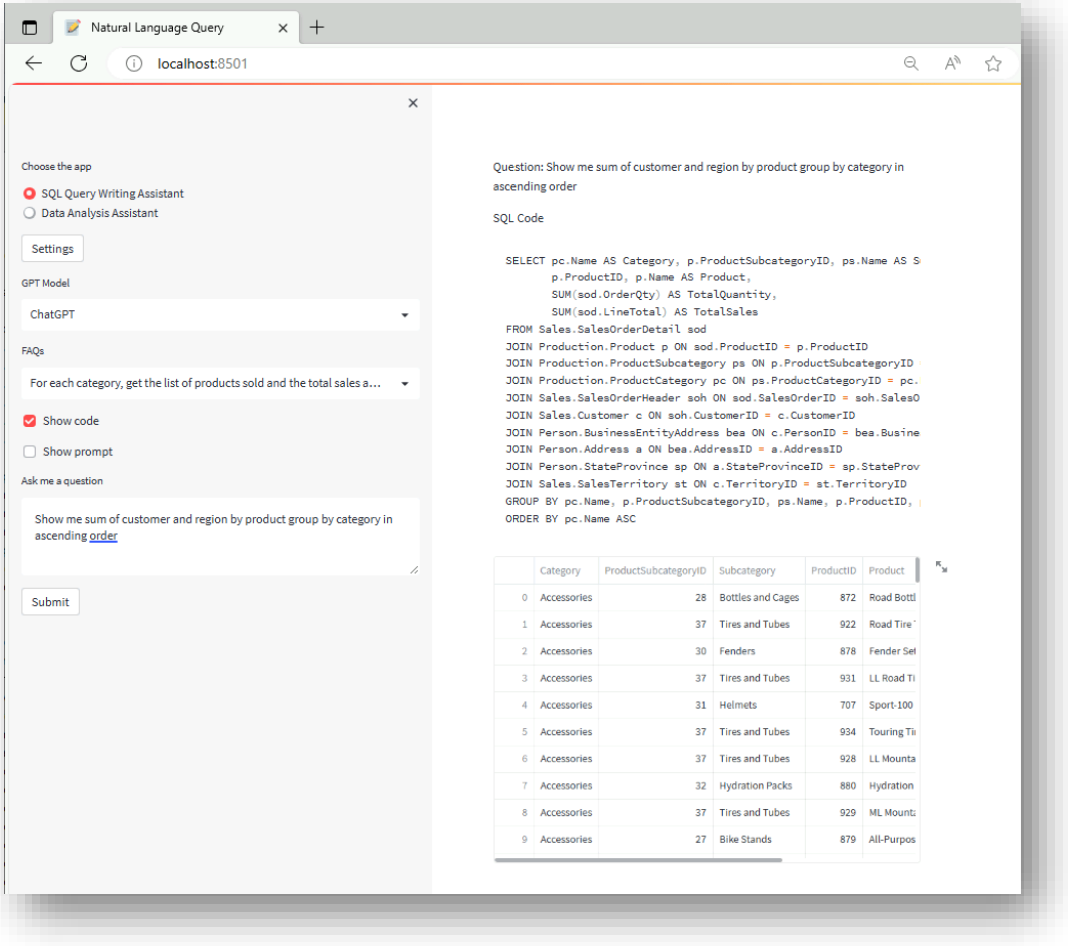
The reason is due to the **object naming** in the database. We purposely changed [Sales].Customer to [Sales].CST. The Azure OpenAI model converts the user query to T-SQL (Transact-SQL). However, since no table with the name "Customer" exists, it cannot query from the table.



You can also test in the Data Analysis Assistant: *Which customers are most likely to churn?* This will also throw an error since it also refers to the customer table.		
FIX: In SSMS, create a **synonym** "sales.customer" for the "sales.cst" table. Within your TEAMXX_LocalMasterDataDB, open a new query window and run the following T-SQL statement: *CREATE SYNONYM [Sales].[Customer]* *FOR [Sales].[Cst]* *GO* **Review** the newly created synonym as shown in the screenshot.		Learn more about synonyms in our documentation: CREATE SYNONYM (Transact-SQL) - SQL Server | Microsoft Learn

Execute the T-sql query:: *Select * from [Sales].[Customer]*		

Submit the question again and view the result:



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***		

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:

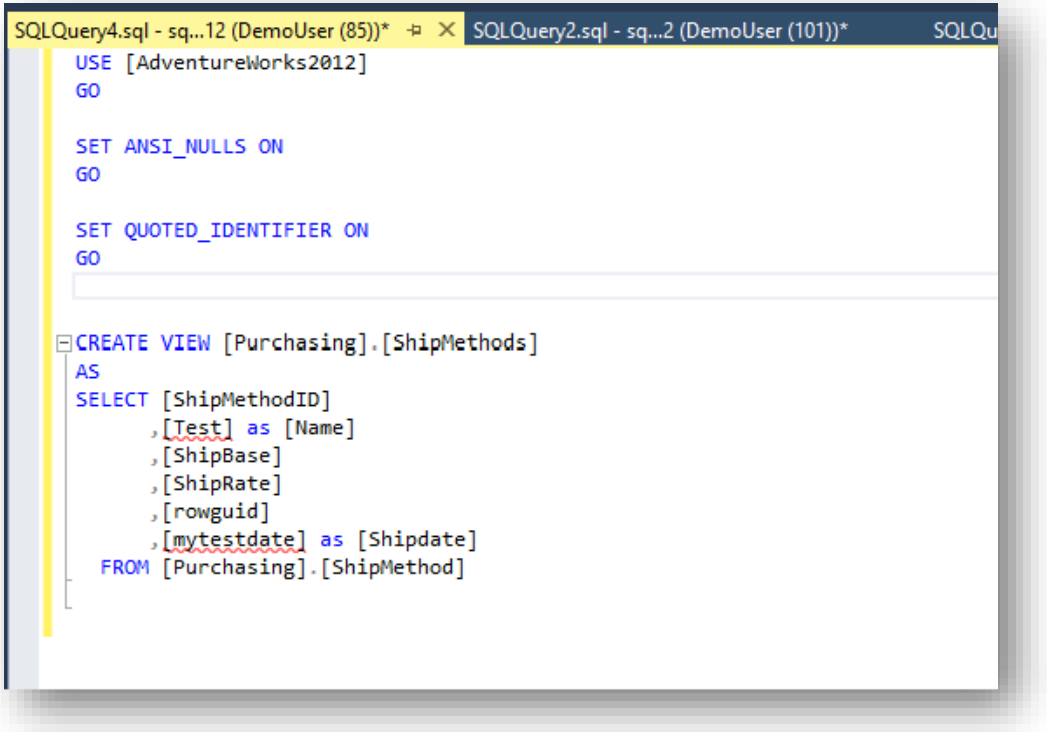
This should group the results by `soh.ShipMethodID` instead of `sm.Name`.

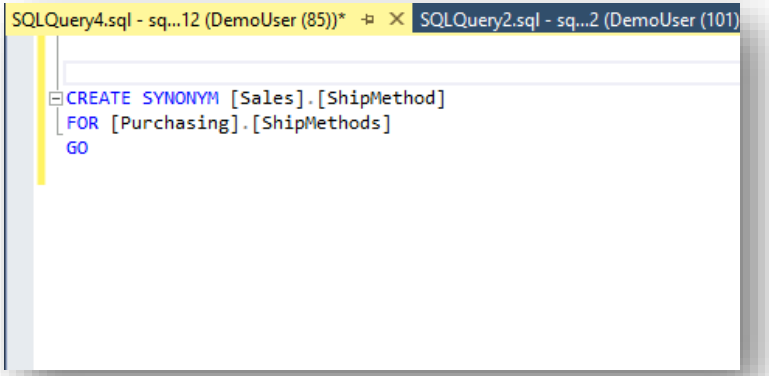
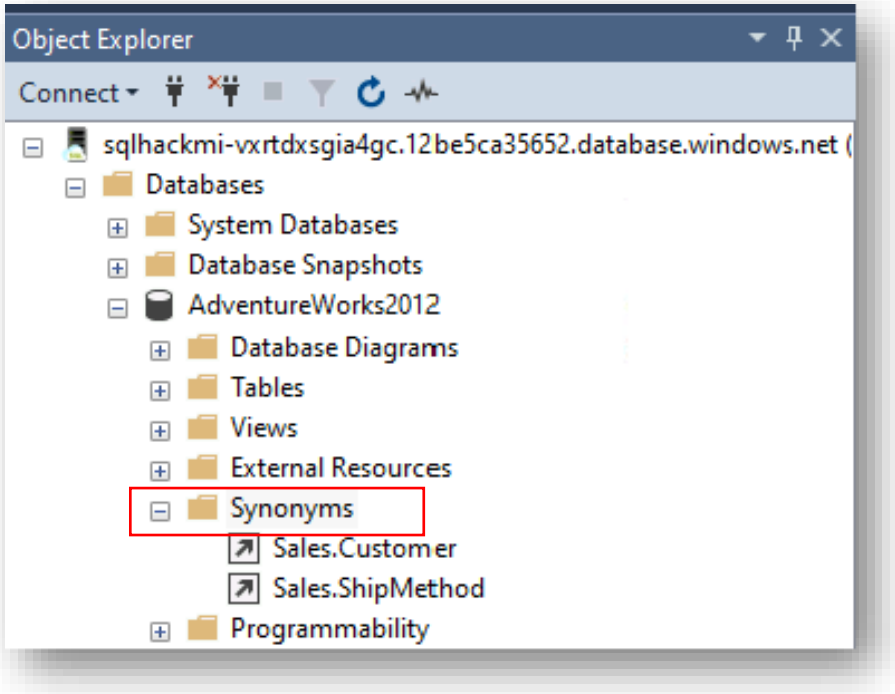
	ProductName	OrderYear	Q1	Q2	Q3	Q4	ShipMethodID
0	All-Purpose Bike Stand	2,013	0	6	65	65	1
1	All-Purpose Bike Stand	2,014	66	47	0	0	1
2	AWC Logo Cap	2,011	0	40	240	265	5
3	AWC Logo Cap	2,012	326	674	608	440	5
4	AWC Logo Cap	2,013	0	54	474	544	1
5	AWC Logo Cap	2,013	514	854	846	482	5
6	AWC Logo Cap	2,014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

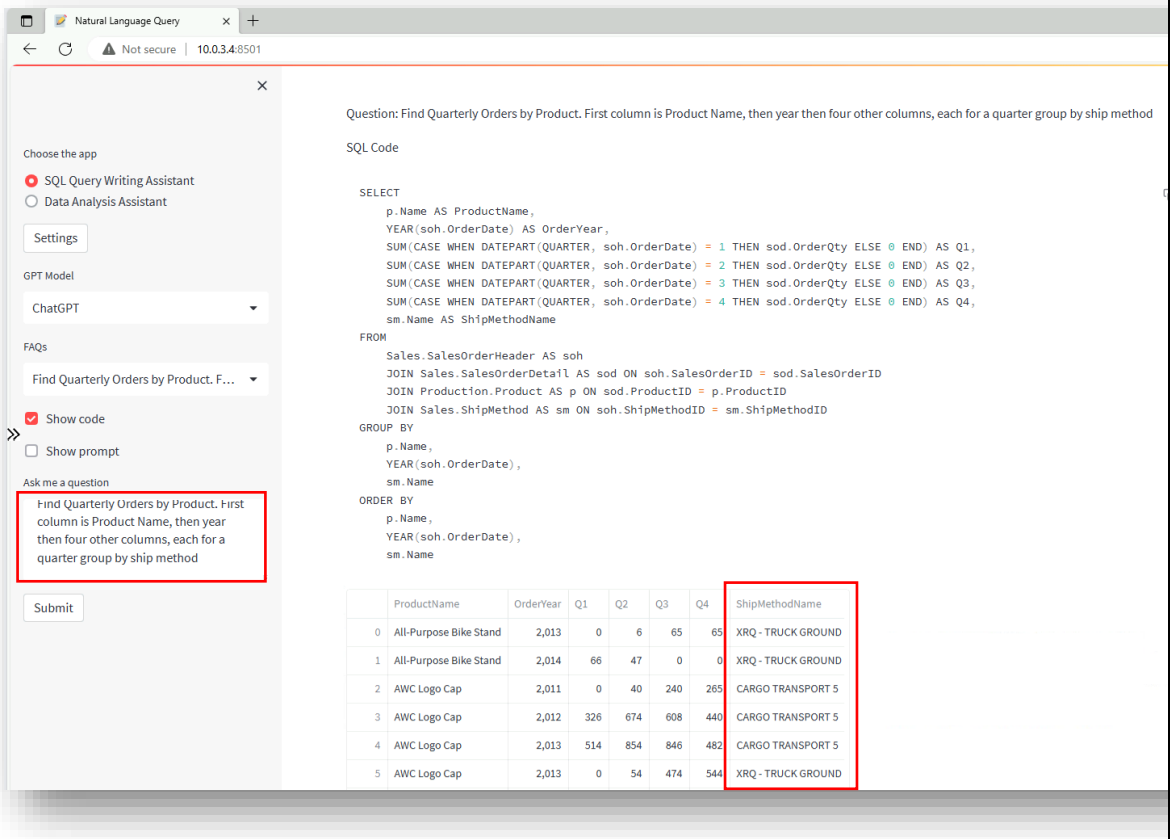
In SSMS, **create a View** via t-sql (adjust to your team DB):

```
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]
```



Create a synonym: CREATE SYNONYM [Sales].[ShipMethod] FOR [Purchasing].[ShipMethods] GO		
Review the synonym		

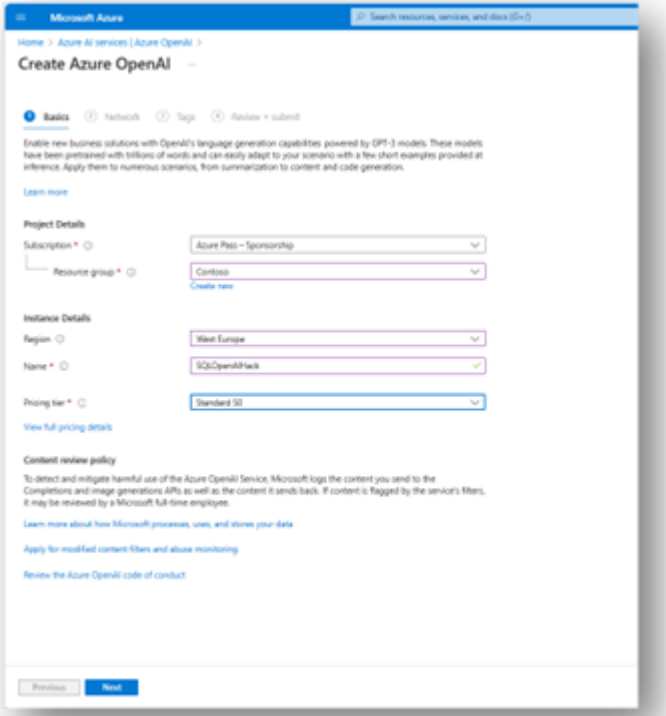
Submit the question again and view the result ***Find Quarterly Orders by Product. First column is Product Name, then year then four other columns, each for a quarter group by ship method***		

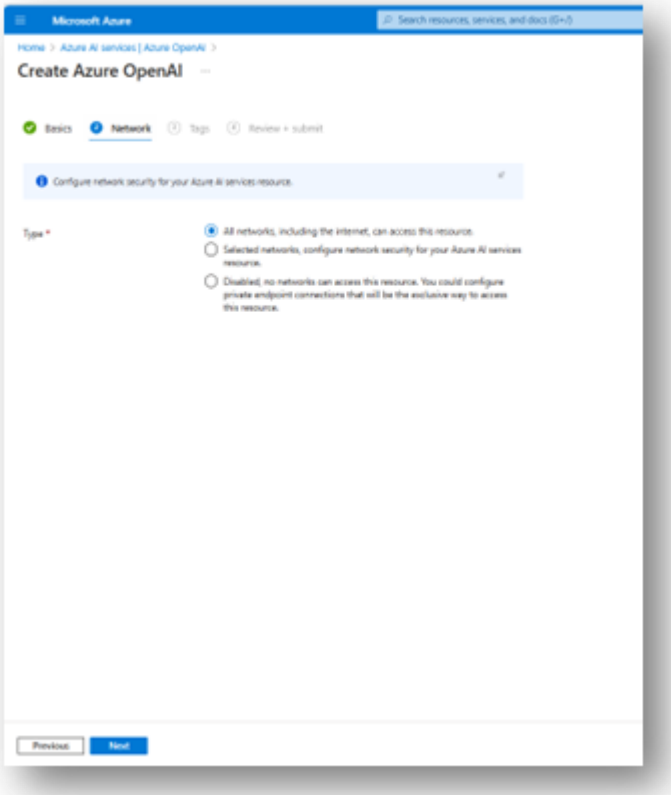
Appendix

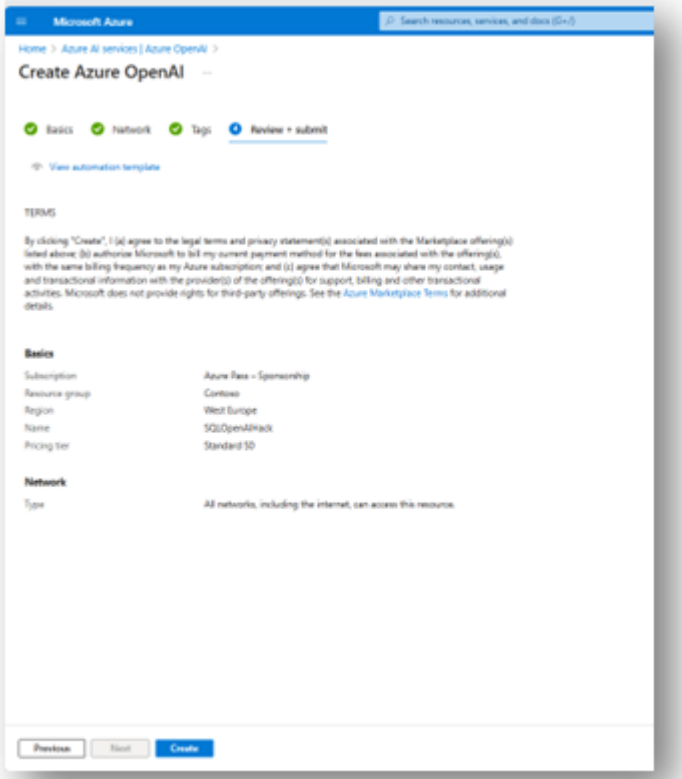
(OPTIONAL) Manual Deployment: You can follow the instructions below if you want to manually deploy the Azure OpenAI resource as well as the environment setup including Visual Studio Code, Git, Python, etc.

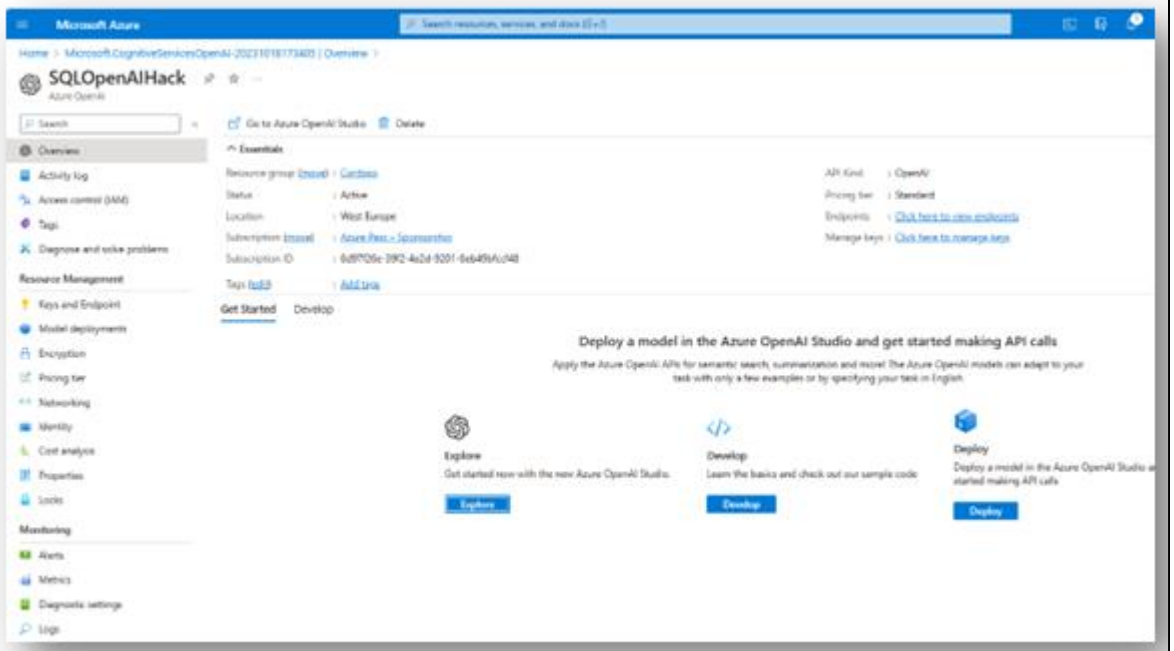
Azure OpenAI Service

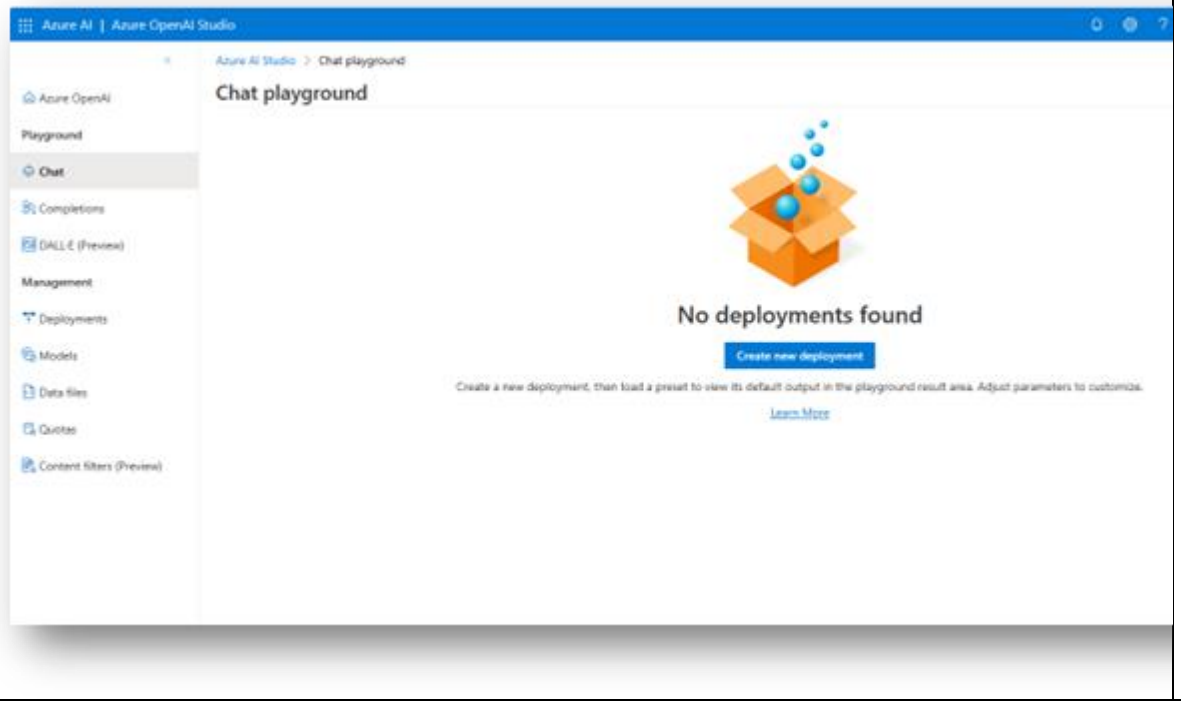
Create an Azure OpenAI resource in an Azure subscription with a **GPT-35-Turbo** deployment and preferably a **GPT-4** deployment. Here we provide options for using both, but GPT-4 should be used to address more difficult & vague questions. We assume that your GPT-4 and ChatGPT deployments are in the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

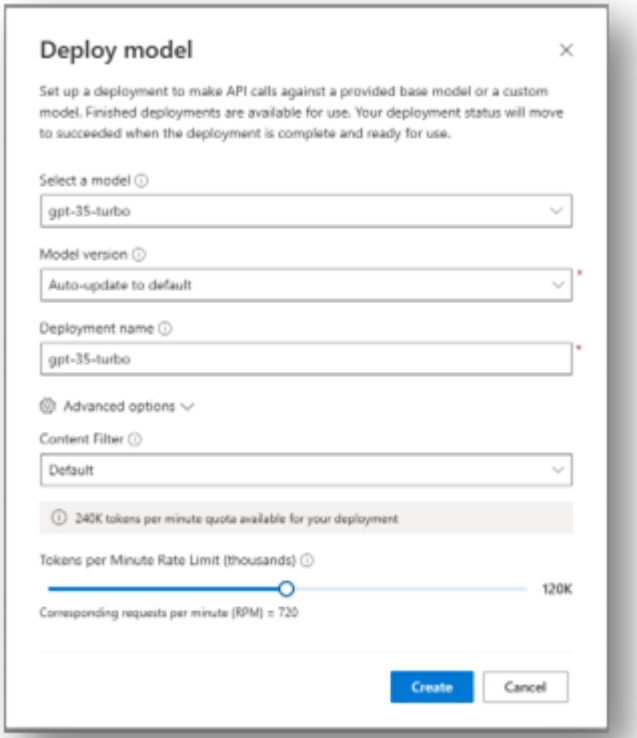
Narrative	Screenshot	Notes
See documentation for instructions: How-to: Create and deploy an Azure OpenAI Service resource - Azure OpenAI | Microsoft Learn		

	 Microsoft Azure 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 Home > Azure AI services | Azure OpenAI > Create Azure OpenAI Basics Network Tags Review + submit View automation template TERMS Basics Subscription Azure Pass – Sponsorship Resource group Contoso Region West Europe Name SQLOpenAIHack Pricing tier Standard S0 Network Type Previous Next Create	

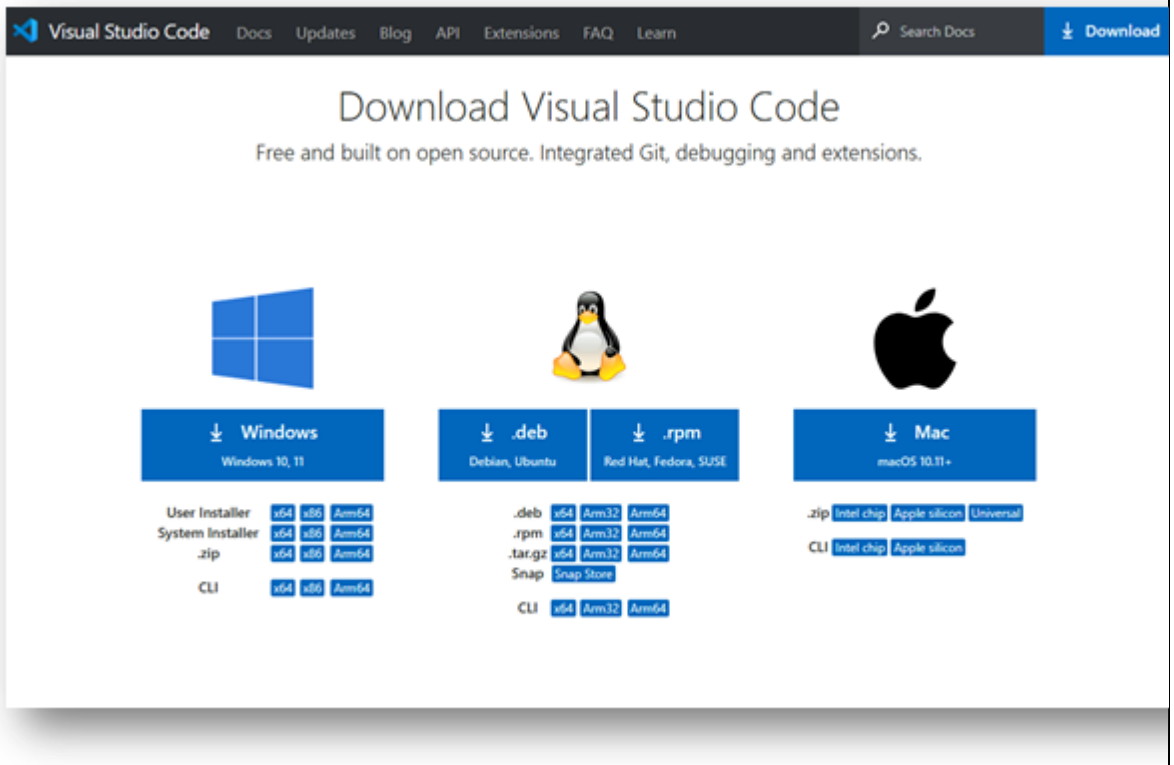
	 Microsoft Azure SQLOpenAIHack Azure OpenAI Overview Activity log Access control (IAM) Tags Diagnose and solve problems Resource Management Keys and Endpoint Model deployments Encryption Pricing tier Networking Identity Cost analysis Properties Locks Monitoring Alerts Metrics Diagnostic settings Logs Go to Azure OpenAI Studio Delete Essentials Get Started Develop Deploy a model in the Azure OpenAI Studio and get started making API calls Explore Develop Deploy	

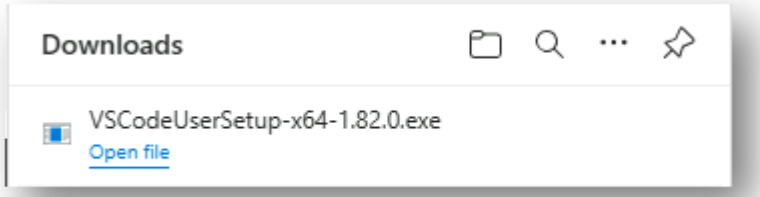
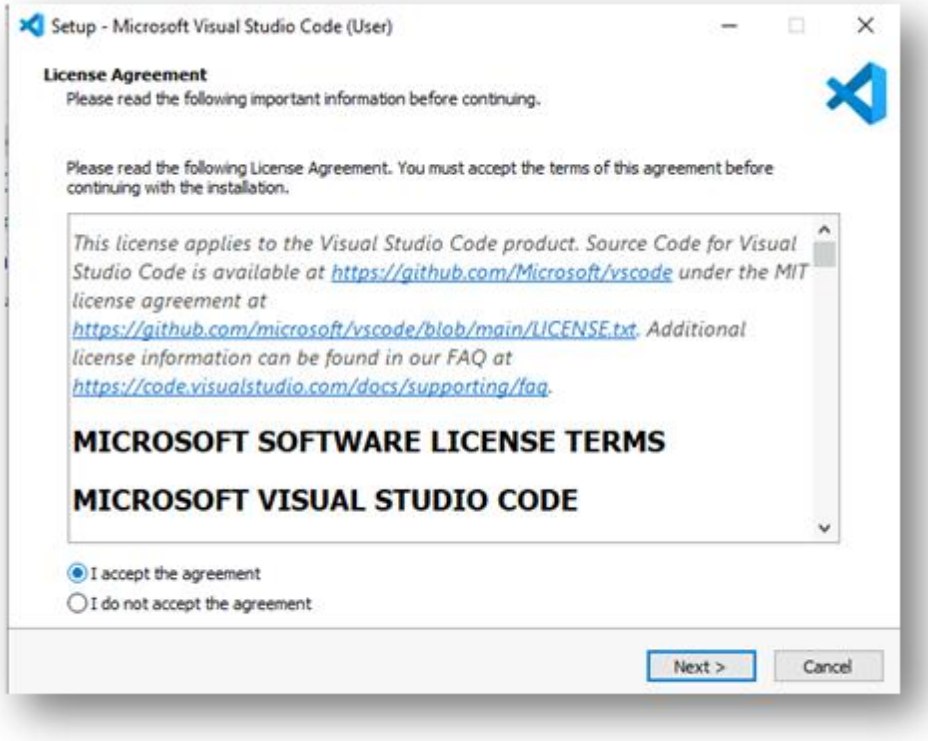

Azure AI | Azure OpenAI Studio
Azure AI Studio > Chat playground
Chat playground
Azure OpenAI
Playground
Chat
Completions
DALL-E (Preview)
Management
Deployments
Models
Data files
Quotes
Content filters (Preview)
No deployments found
Create new deployment
Learn More

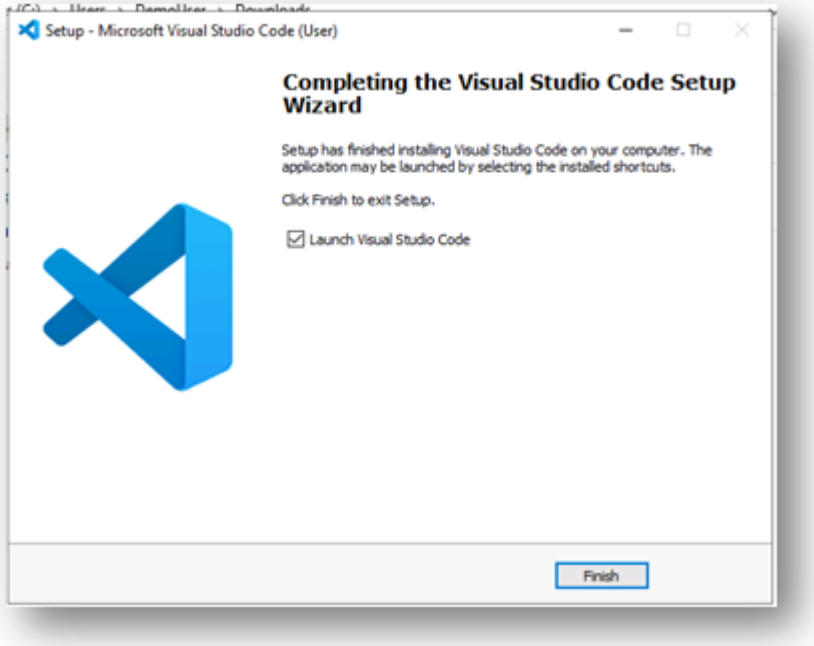

Deploy model
Set up a deployment to make API calls against a provided base model or a custom model. Finished deployments are available for use. Your deployment status will move to succeeded when the deployment is complete and ready for use.
Select a model
gpt-35-turbo
Model version
Auto-update to default
Deployment name
gpt-35-turbo
Advanced options
Content Filter
Default
240K tokens per minute quota available for your deployment
Tokens per Minute Rate Limit (thousands)
120K
Corresponding requests per minute (RPM) = 720
Create
Cancel

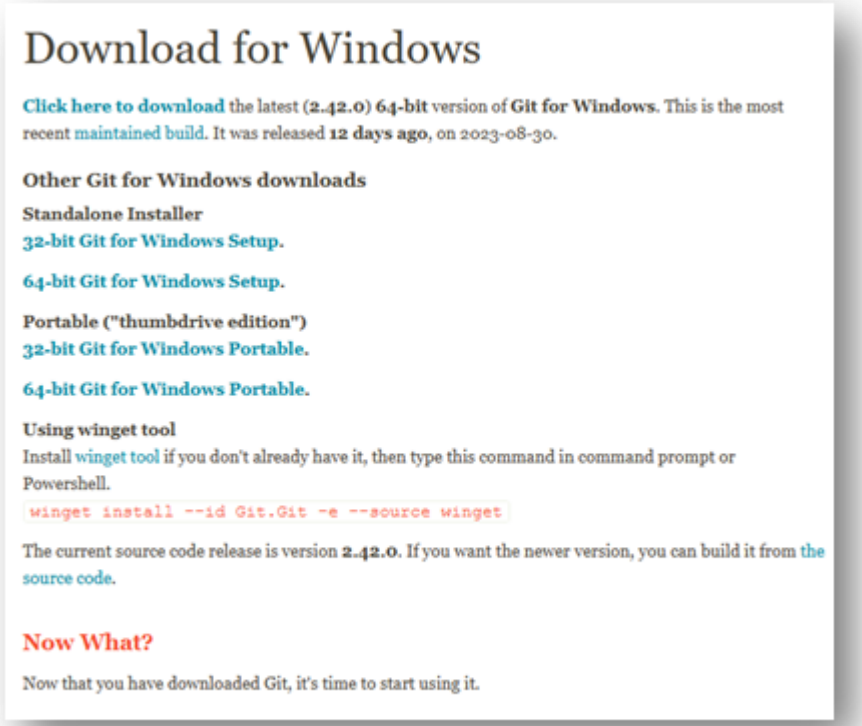
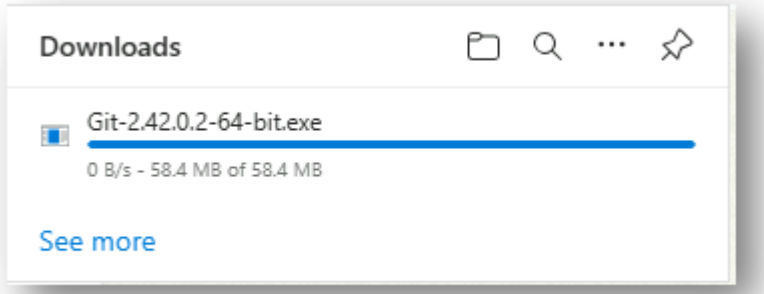
Environment Setup

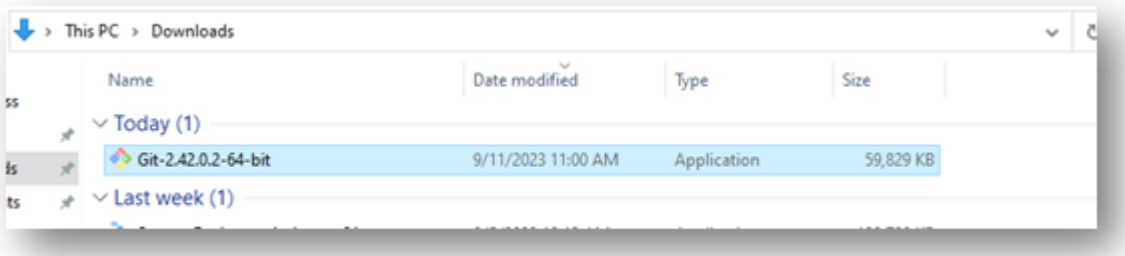
The environment has been provisioned for you upfront for this workshop. If you would like to manually set up the environment on your local machine, you can follow the instructions:

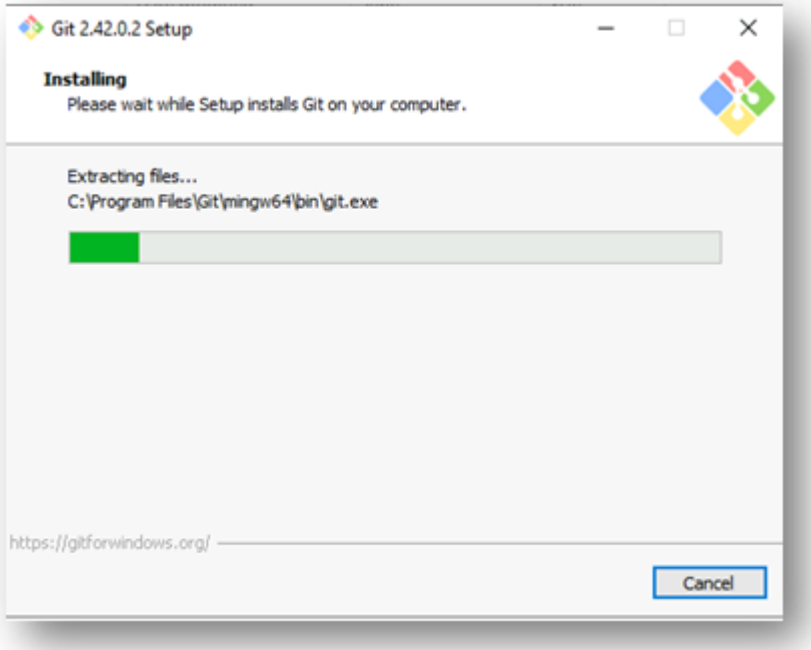
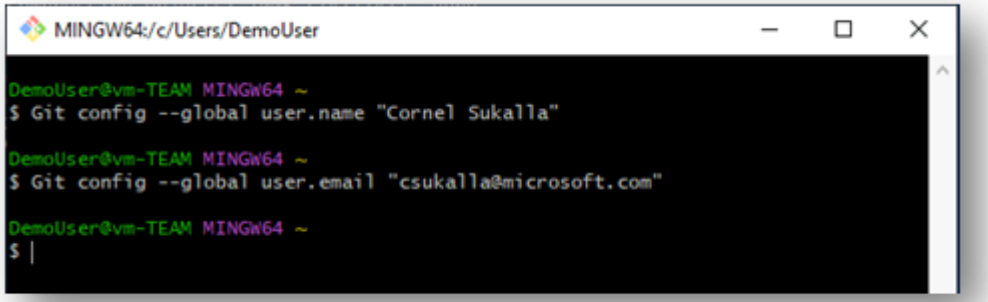
Narrative	Screenshot	Notes
Download Visual Studio Code https://code.visualstudio.com/download		

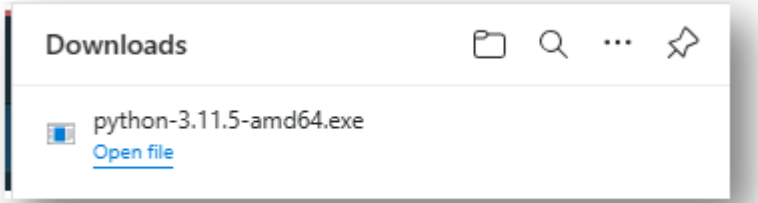
		
Install Visual Studio Code		

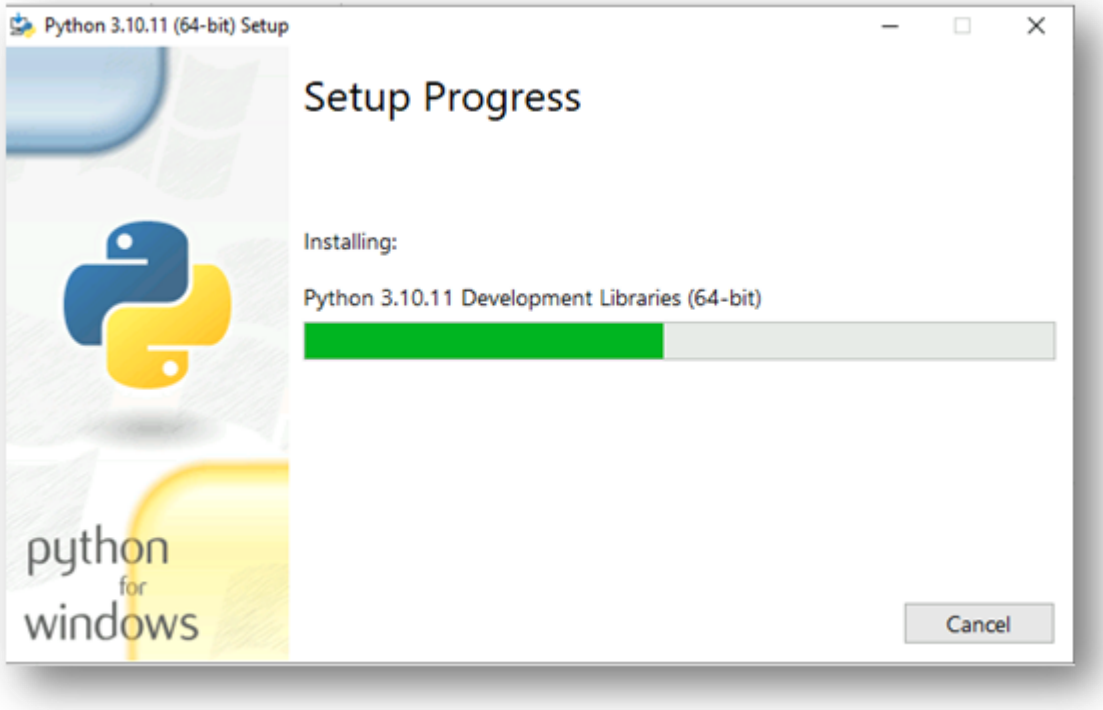
	 Setup - Microsoft Visual Studio Code (User) Completing the Visual Studio Code Setup Wizard Setup has finished installing Visual Studio Code on your computer. The application may be launched by selecting the installed shortcuts. Click Finish to exit Setup. Launch Visual Studio Code Finish	

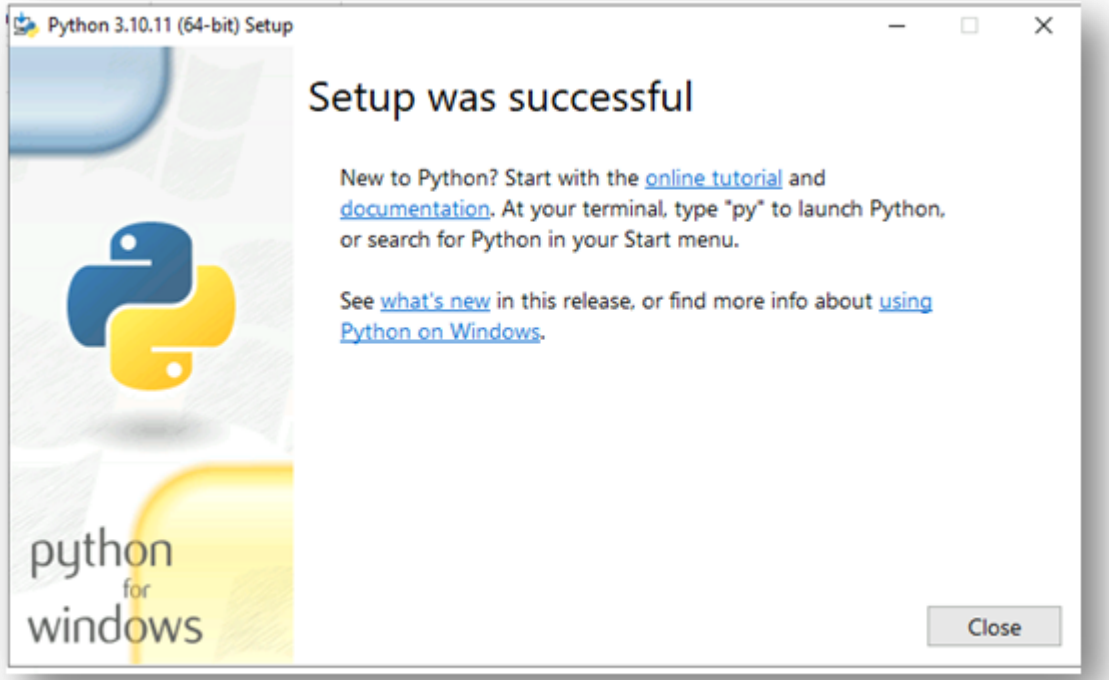
Download Git: https://git-scm.com/download/win	 	

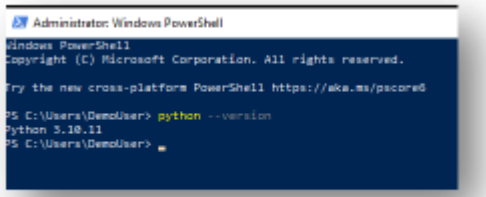
	 This PC › Downloads Name Date modified Type Size Today (1) Git-2.42.0.2-64-bit 9/11/2023 11:00 AM Application 59,829 KB Last week (1)	
Install GIT	 Git 2.42.0.2 Setup Information Please read the following important information before continuing. When you are ready to continue with Setup, click Next. GNU General Public License Version 2, June 1991 Copyright (C) 1989, 1991 Free Software Foundation, Inc. 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed. Preamble The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change https://gitforwindows.org/ Next Cancel	

		
(Optional) Set Git Variables		

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.		


Python 3.10.11 (64-bit) Setup
Setup was successful
New to Python? Start with the online tutorial and documentation. At your terminal, type "py" to launch Python, or search for Python in your Start menu.
See what's new in this release, or find more info about using Python on Windows.
python for windows
Close


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