Explore Azure Synapse Studio

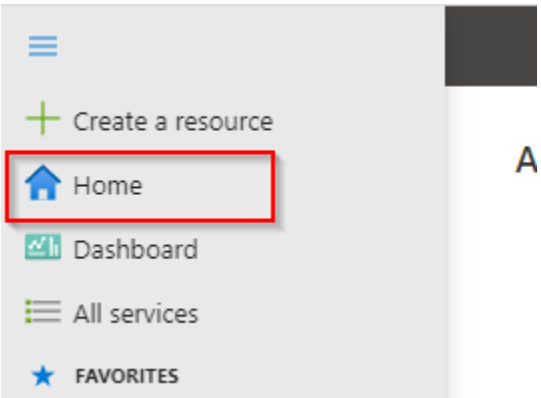
Duration: 90 minutes

The main task for this exercise is as follows:

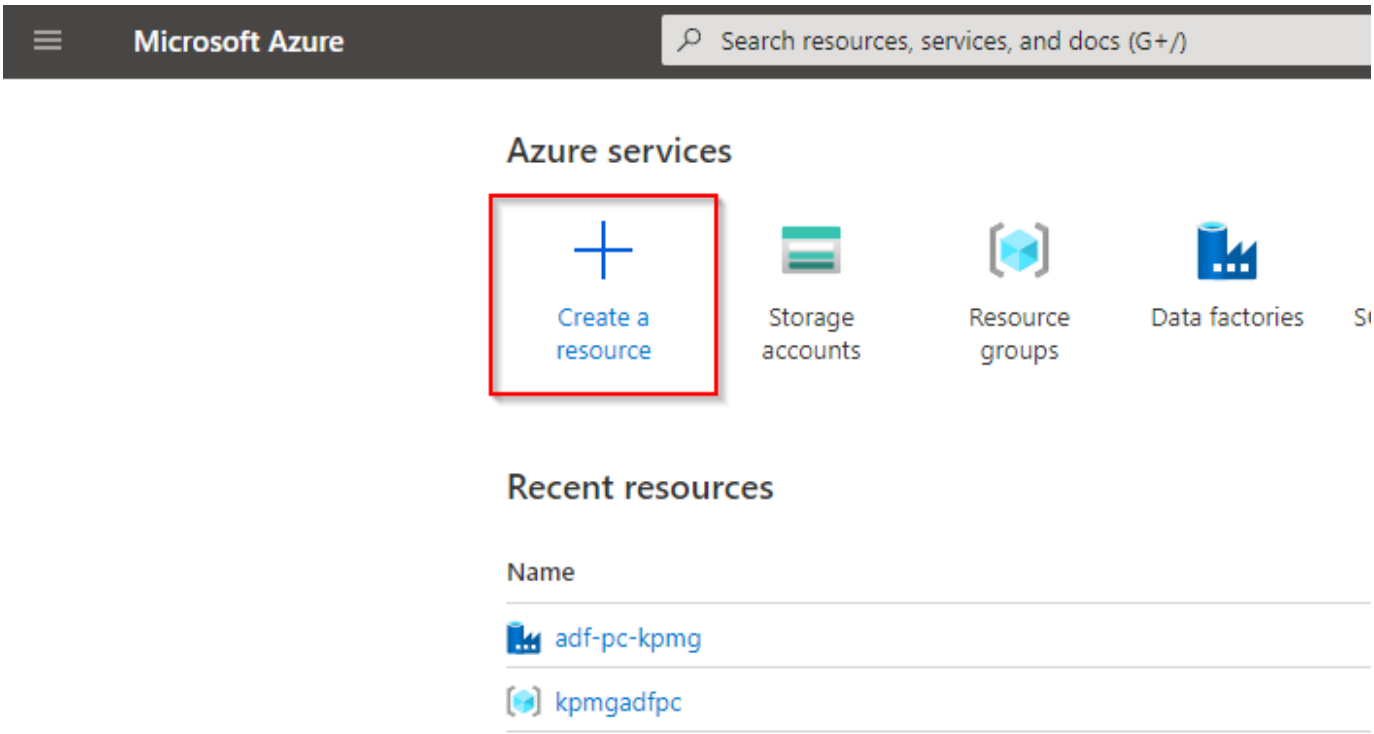
1. Provision an Azure Storage Account (Data Lake Gen2)
2. Provision an Azure Synapse Analytics workspace
3. Explore Synapse Studio
4. Ingest data with a pipeline
5. Use a serverless SQL pool to analyze data
6. Use a Spark pool to analyze data

Task 1: Provision an Azure Storage Account (Data Lake Gen2)

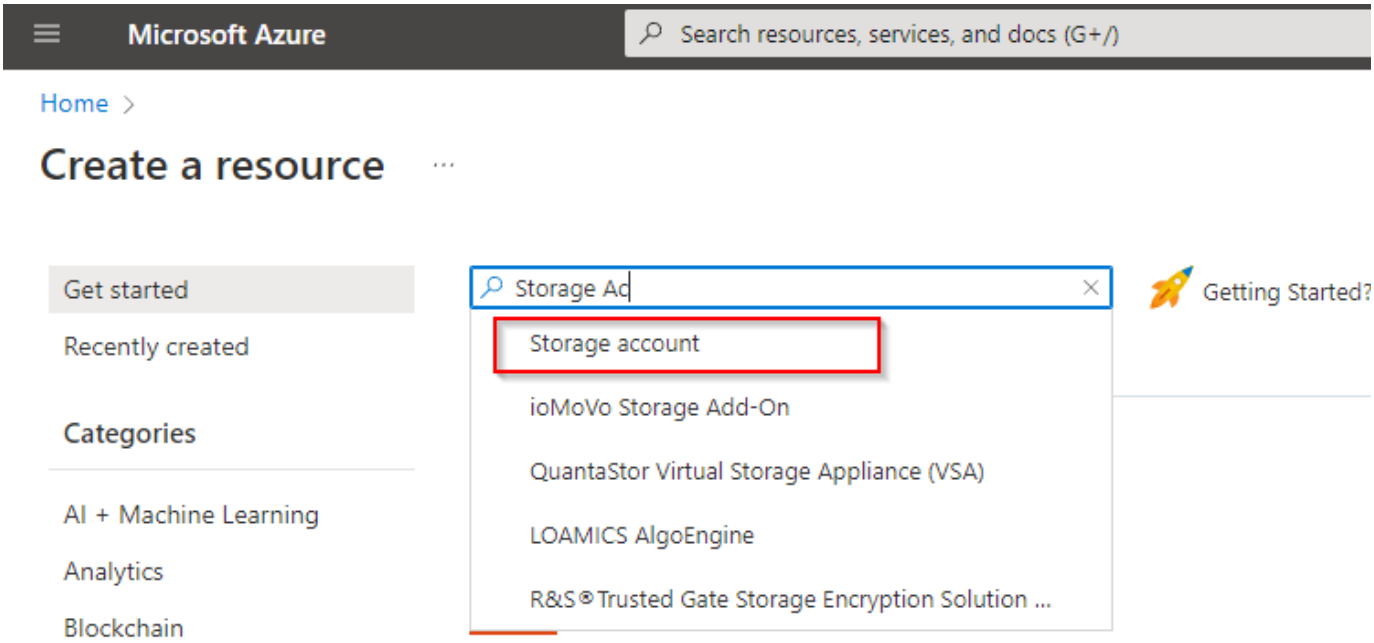
1. In the Azure portal, at the top left of the screen, click on the **Home** hyperlink



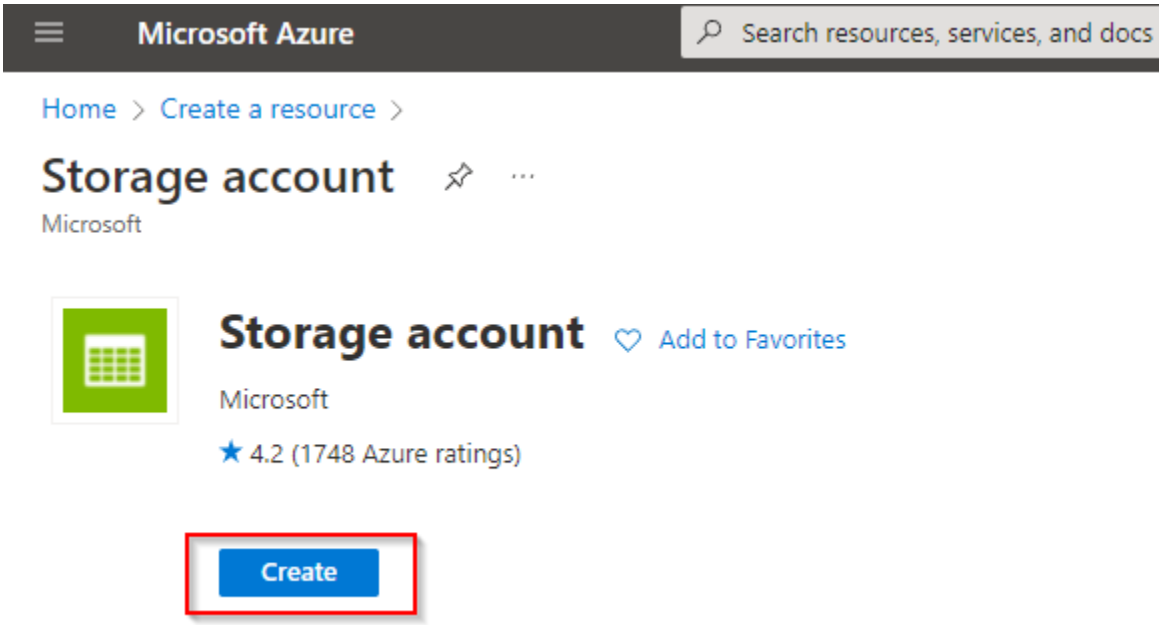
2. In the Azure portal, click on the **+ Create a resource** icon.



3. In the New screen, click in the **Search services and marketplace** text box, and type the word **storage account**. Click **Storage account** in the list that appears.



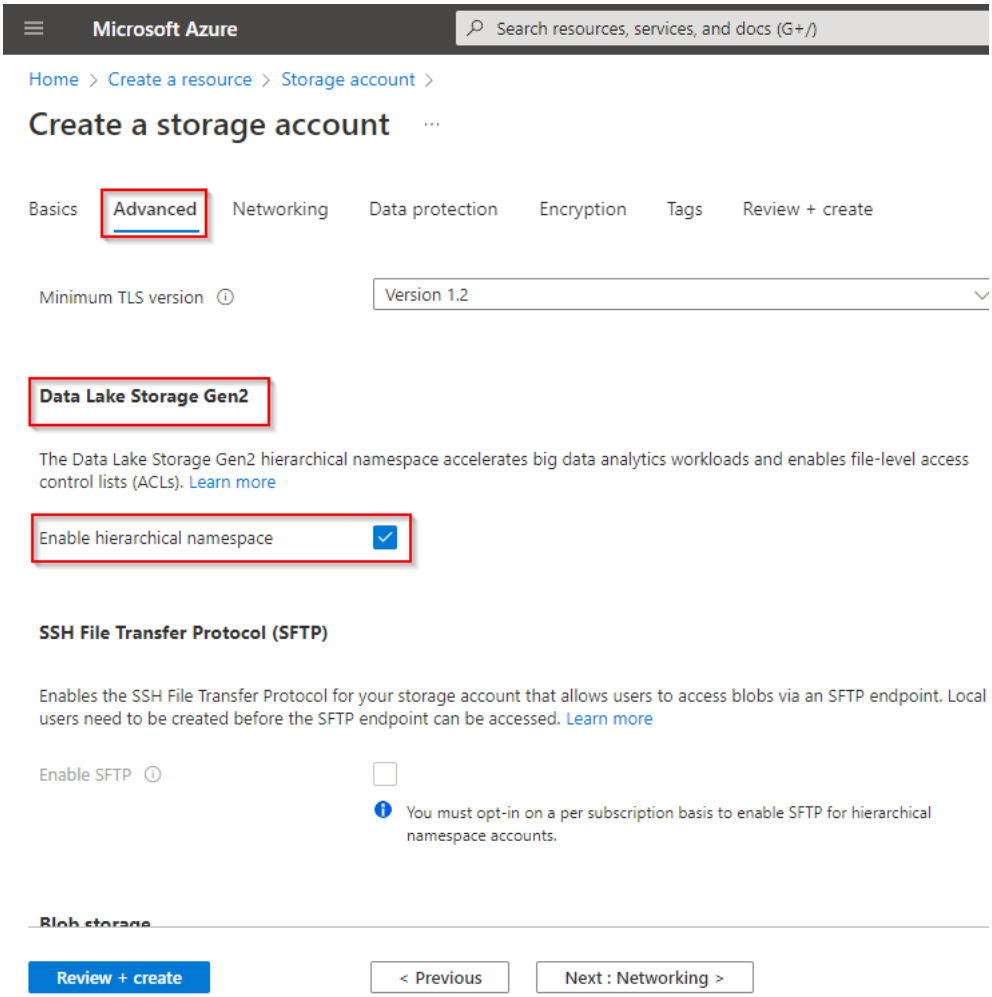
4. In the **Storage account** screen, click **Create**.



5. From the **Create a storage account** screen in the **Basics** tab, create the first storage account with the following settings:
    - Under the project details, specify the following settings:
        - **Subscription**: the name of the subscription you are using in this lab
        - **Resource group**: **synapse-xx-rg**, where **xx** are your initials.
    - Under the Instance details, specify the following settings:
        - **Storage account name**: **asastoragexx**, where **xx** are your initials.

- **Region**: the name of the Azure region which is closest to the lab location.
- **Performance**: **Standard**.
- **Redundancy**: **Locally-redundant storage (LRS)**
- Select **Make read access to data available in the event of regional unavailability.**

- Select **Advanced** tab in create storage account
  - Under Data Lake Storage Gen2 details
  - Enable **hierarchical namespace**: mark is Select

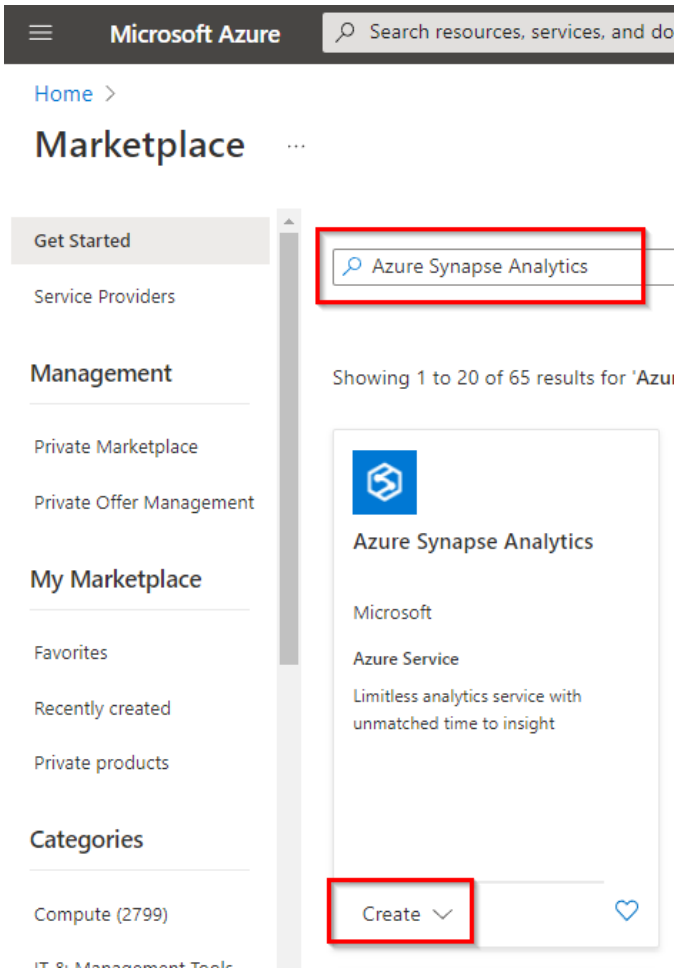


6. In the **Create storage account** screen, click **Review + create**.
7. After the validation of the **Create storage account*** screen, click **Create**.
8. Once the storage account created create click on **Go to resource**, under **Data storage** blade click on **Containers** then click on **+ Container** and give name **files** then click on Create button.

Task 2: Provision an Azure Synapse Analytics

Create your azure synapse workspace: Use the Azure Portal to create your synapse workspace.

1. In browser, go to the Azure portal tab, click on the **+ Create a resource** icon, type **synapse**, and then click **Azure Synapse Analytics** from the resulting search, and then click **Create**.



2. In the New Synapse screen, create a new **Azure Synapse Analytics** with the following options:

   Basic Tab

   - **Subscription**: Your subscription
   - **Resource group**: synapse-xx-rg (xx is your initials)
   - **Managed resource group**: Leave empty
   - **Name**: synapse-xx, where xx are your initials
   - **Region**: westus or centralindia
   - **Select Data Lake Storage Gen2**: From subscription
     - **Account name**: from drop down list select **asastoragexx** (where xx are your initials)

        - **File system name**: from drop down list select or enter **files**
    - Leave other options to their default settings

Security Tab

- **SQL Password**: P@55w.rd123
- **Confirm password**: P@55w.rd123

Leave rest of the setting to default

Click **Review + Create** and then create.

Wait for few minutes for the deployment successfully complete.

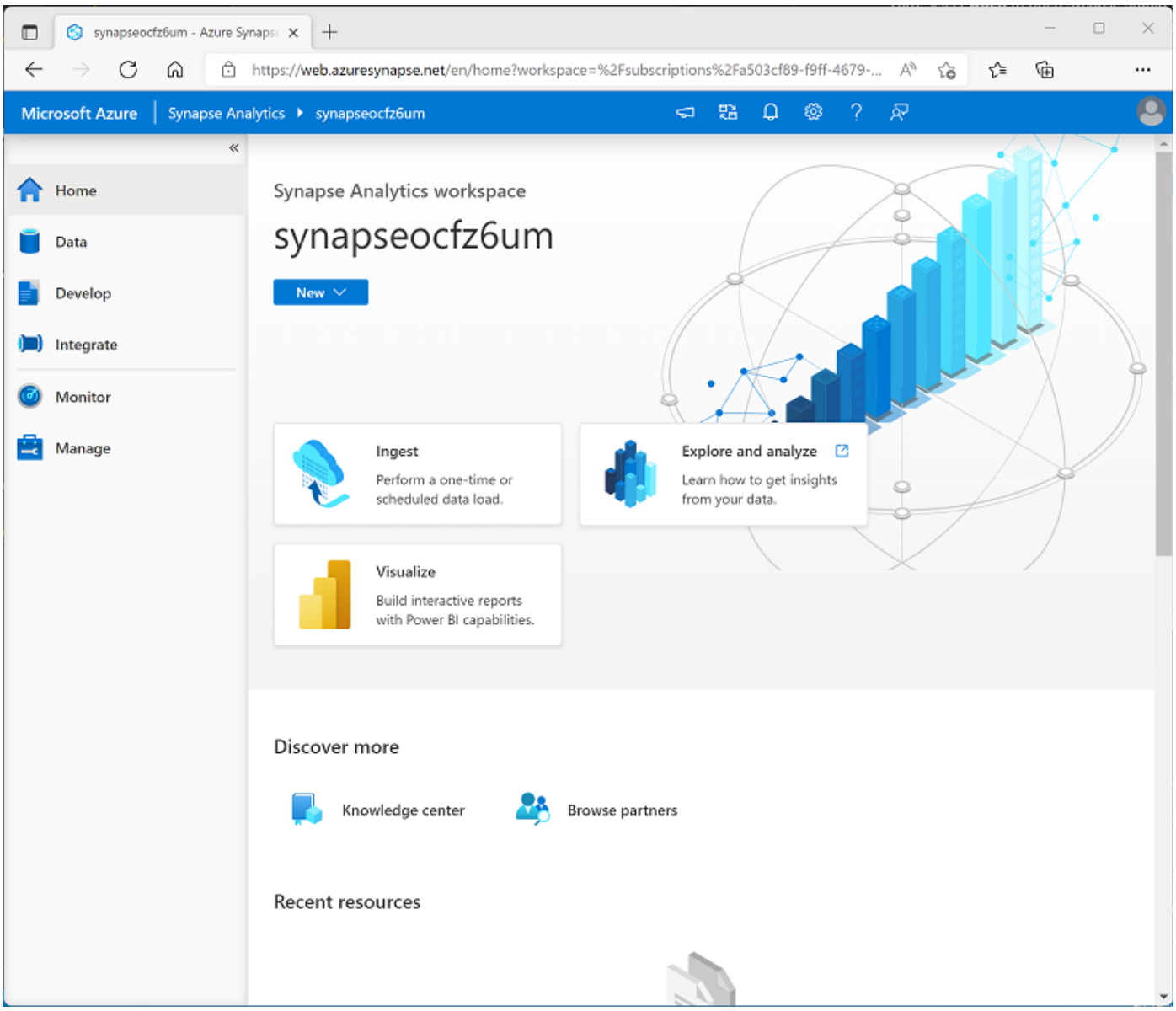
Task 3: Explore Synapse Studio

Synapse Studio is a web-based portal in which you can manage and work with the resources in your Azure Synapse Analytics workspace.

1. When the setup script has finished running, in the Azure portal, go to the **synpase-xx-rg** resource group that it created, and notice that this resource group contains your Synapse workspace & a Storage account for your data lake.

2. Select your Synapse workspace, and in its **Overview** page, in the **Open Synapse Studio** card, select **Open** to open Synapse Studio in a new browser tab.

3. On the left side of Synapse Studio, use the **››** icon to expand the menu - this reveals the different pages within Synapse Studio that you'll use to manage resources and perform data analytics tasks, as shown here:



4. View the **Data** page, and note that there are two tabs containing data sources:
    - A **Workspace** tab containing databases defined in the workspace.
    - A **Linked** tab containing data sources that are linked to the workspace, including Azure Data Lake storage.
5. View the **Develop** page, which is currently empty. This is where you can define scripts and other assets used to develop data processing solutions.
6. View the **Integrate** page, which is also empty. You use this page to manage data ingestion and integration assets; such as pipelines to transfer and transform data between data sources.
7. View the **Monitor** page. This is where you can observe data processing jobs as they run and view their history.
8. View the **Manage** page. This is where you manage the pools, runtimes, and other assets used in your Azure Synapse workspace. View each of the tabs in the **Analytics pools** section and note that your workspace includes the following pools:
    - **SQL pools**:
        - **Built-in**: A *serverless* SQL pool that you can use on-demand to explore or process data in a data lake by using SQL commands.
        - A *dedicated* SQL pool that hosts a relational data warehouse database.
    - **Apache Spark pools**:
        - You can use on-demand to explore or process data in a data lake by using programming languages like Scala or Python.
    - **Data Explorer pools**:
        - A Data Explorer pool that you can use to analyze data by using Kusto Query Language (KQL).

Task 4: Ingest data with a pipeline

Use the Copy Data task to create a pipeline

1. In Synapse Studio, on the **Home** page, select **Ingest** to open the **Copy Data** tool

2. In the Copy Data tool, on the **Properties** step, ensure that **Built-in copy task** and **Run once now** are selected, and click **Next >**.

3. On the **Source** step, in the **Dataset** substep, select the following settings:
   - **Source type**: All
   - **Connection**: *Create a new connection, and in the **Linked service** pane that appears, on the **File** tab, select **HTTP**. Then continue and create a connection to a data file using the following settings:*
     - **Name**: Products
     - **Description**: Product list via HTTP
     - **Connect via integration runtime**: AutoResolveIntegrationRuntime
     - **Base URL**: https://raw.githubusercontent.com/MicrosoftLearning/mslearn-synapse/master/Allfiles/Labs/01/adventureworks/products.csv
     - **Server Certificate Validation**: Enable
     - **Authentication type**: Anonymous

4. After creating the connection, on the **Source data store** page, ensure the following settings are selected, and then select **Next >**:
   - **Relative URL**: *Leave blank*
   - **Request method**: GET
   - **Additional headers**: *Leave blank*
   - **Binary copy**: Unselected
   - **Request timeout**: *Leave blank*
   - **Max concurrent connections**: *Leave blank*

5. On the **Source** step, in the **Configuration** substep, select **Preview data** to see a preview of the product data your pipeline will ingest, then close the preview.

6. After previewing the data, on the **File format settings** page, ensure the following settings are selected, and then select **Next >**:
   - **File format**: DelimitedText
   - **Column delimiter**: Comma (,)
   - **Row delimiter**: Line feed (\n)
   - **First row as header**: Selected
   - **Compression type**: None

7. On the **Destination** step, in the **Dataset** substep, select the following settings:
    - **Destination type**: Azure Data Lake Storage Gen 2
    - **Connection**: *Select the existing connection to your data lake store (this was created for you when you created the workspace).*

8. After selecting the connection, on the **Destination/Dataset** step, ensure the following settings are selected, and then select **Next >**:
    - **Folder path**: files/product_data
    - **File name**: products.csv
    - **Copy behavior**: None
    - **Max concurrent connections**: *Leave blank*
    - **Block size (MB)**: *Leave blank*

9. On the **Destination** step, in the **Configuration** substep, on the **File format settings** page, ensure that the following properties are selected. Then select **Next >**:
    - **File format**: DelimitedText
    - **Column delimiter**: Comma (,)
    - **Row delimiter**: Line feed (\n)
    - **Add header to file**: Selected
    - **Compression type**: None
    - **Max rows per file**: *Leave blank*
    - **File name prefix**: *Leave blank*

10. On the **Settings** step, enter the following settings and then click **Next >**:
    - **Task name**: Copy products
    - **Task description** Copy products data
    - **Fault tolerance**: *Leave blank*
    - **Enable logging**: Unselected
    - **Enable staging**: Unselected

11. On the **Review and finish** step, on the **Review** substep, read the summary and then click **Next >**.

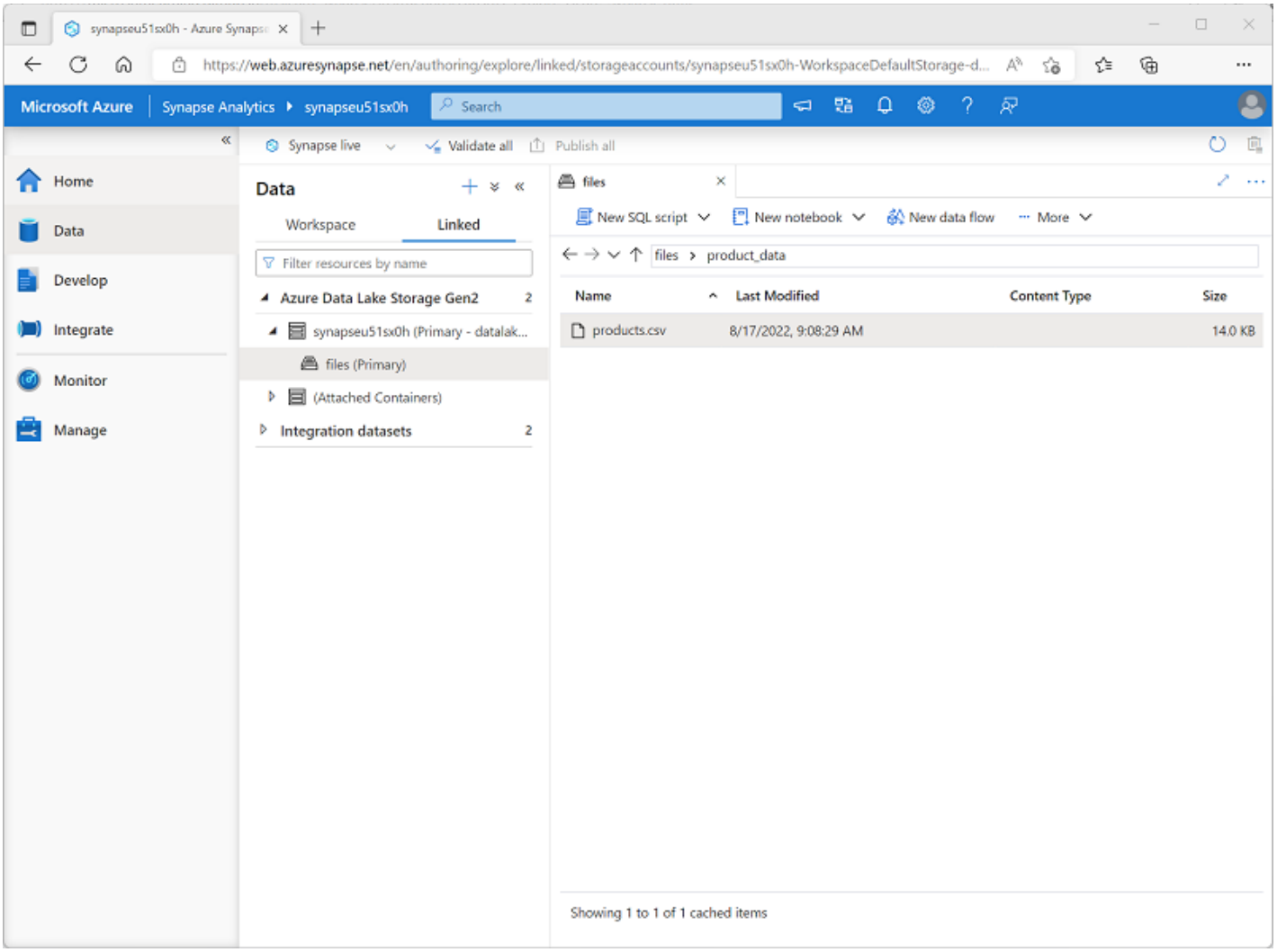
12. On the **Deployment** step, wait for the pipeline to be deployed and then click **Finish**.

13. In Synapse Studio, select the **Monitor** page, and in the **Pipeline runs** tab, wait for the **Copy products** pipeline to complete with a status of **Succeeded** (you can use the ↻ **Refresh** button on the Pipeline runs page to refresh the status).

14. View the **Integrate** page, and verify that it now contains a pipeline named **Copy products**.

View the ingested data

1. On the **Data** page, select the **Linked** tab and expand the **Product Files** hierarchy until you see the **files** file storage for your Synapse workspace. Then select the file storage to verify that a folder named **product_data** containing a file named **products.csv** has been copied to this location, as shown here:



2. Right-click the **products.csv** data file and select **Preview** to view the ingested data. Then close the preview.

Task 5: Use a serverless SQL pool to analyze data

One of the most common ways to query data is to use SQL, and in Synapse Analytics you can use a serverless SQL pool to run SQL code against data in a data lake.

1. In Synapse Studio, right-click the **products.csv** file in the file storage for your Synapse workspace, point to **New SQL script**, and select **Select TOP 100 rows**.

2. In the **SQL Script 1** pane that opens, review the SQL code that has been generated, which should be similar to this:

3. -- This is auto-generated code

```
SELECT
   TOP 100 *
FROM
   OPENROWSET(
      BULK 'https://asastoragexx.dfs.core.windows.net/files/product_data/products.csv',
      FORMAT = 'CSV',
      PARSER_VERSION='2.0'
         ) AS [result]
```

   This code opens a rowset from the text file you imported and retrieves the first 100 rows of data.

4. In the **Connect to** list, ensure **Built-in** is selected - this represents the built-in SQL Pool that was created with your workspace.

5. On the toolbar, use the ▷ **Run** button to run the SQL code, and review the results, which should look similar to this:

C1	C2	C3	C4
ProductID	ProductName	Category	ListPrice
771	Mountain-100 Silver, 38	Mountain Bikes	3399.9900
772	Mountain-100 Silver, 42	Mountain Bikes	3399.9900
...	...	...	...

6. Note the results consist of four columns named C1, C2, C3, and C4; and that the first row in the results contains the names of the data fields. To fix this problem, add a HEADER_ROW = TRUE parameters to the OPENROWSET function as shown here (replacing *asastoragexx* with the name of your data lake storage account), and then rerun the query:

```
SELECT
   TOP 100 *
FROM
   OPENROWSET(
      BULK 'https:// asastoragexx.dfs.core.windows.net/files/product_data/products.csv',
      FORMAT = 'CSV',
      PARSER_VERSION='2.0',
      HEADER_ROW = TRUE
   ) AS [result]
```

Now the results look like this:

ProductID	ProductName	Category	ListPrice
771	Mountain-100 Silver, 38	Mountain Bikes	3399.9900
772	Mountain-100 Silver, 42	Mountain Bikes	3399.9900
...	...	...	...

7. Modify the query as follows (replacing *asastoragexx* with the name of your data lake storage account):

```
SELECT
   Category, COUNT(*) AS ProductCount
FROM
   OPENROWSET(
      BULK 'https:// asastoragexx.dfs.core.windows.net/files/product_data/products.csv',
      FORMAT = 'CSV',
      PARSER_VERSION='2.0',
      HEADER_ROW = TRUE
   ) AS [result]
         GROUP BY Category;
```

8. Run the modified query, which should return a resultset that contains the number products in each category, like this:

Category	ProductCount
Bib Shorts	3
Bike Racks	1
...	...

9. In the **Properties** pane for **SQL Script 1**, change the **Name** to **Count Products by Category**. Then in the toolbar, select **Publish** to save the script.

10. Close the **Count Products by Category** script pane.

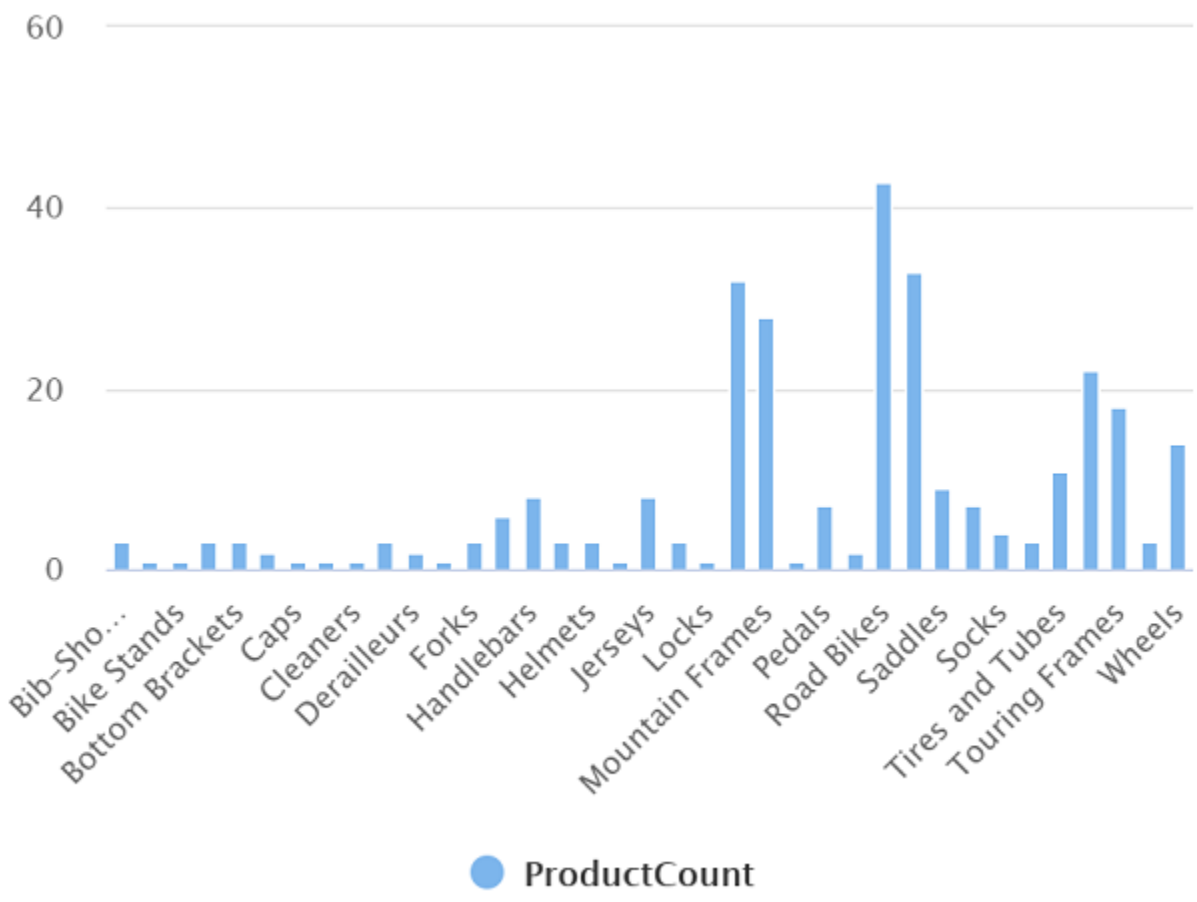
11. In Synapse Studio, select the **Develop** page, and notice that your published **Count Products by Category** SQL script has been saved there.

12. Select the **Count Products by Category** SQL script to reopen it. Then ensure that the script is connected to the **Built-in** SQL pool and run it to retrieve the product counts.

13. In the **Results** pane, select the **Chart** view, and then select the following settings for the chart:

    - **Chart type**: Column
    - **Category column**: Category
    - **Legend (series) columns**: ProductCount
    - **Legend position**: bottom - center
    - **Legend (series) label**: *Leave blank*
    - **Legend (series) minimum value**: *Leave blank*
    - **Legend (series) maximum**: *Leave blank*
    - **Category label**: *Leave blank*

    The resulting chart should resemble this:

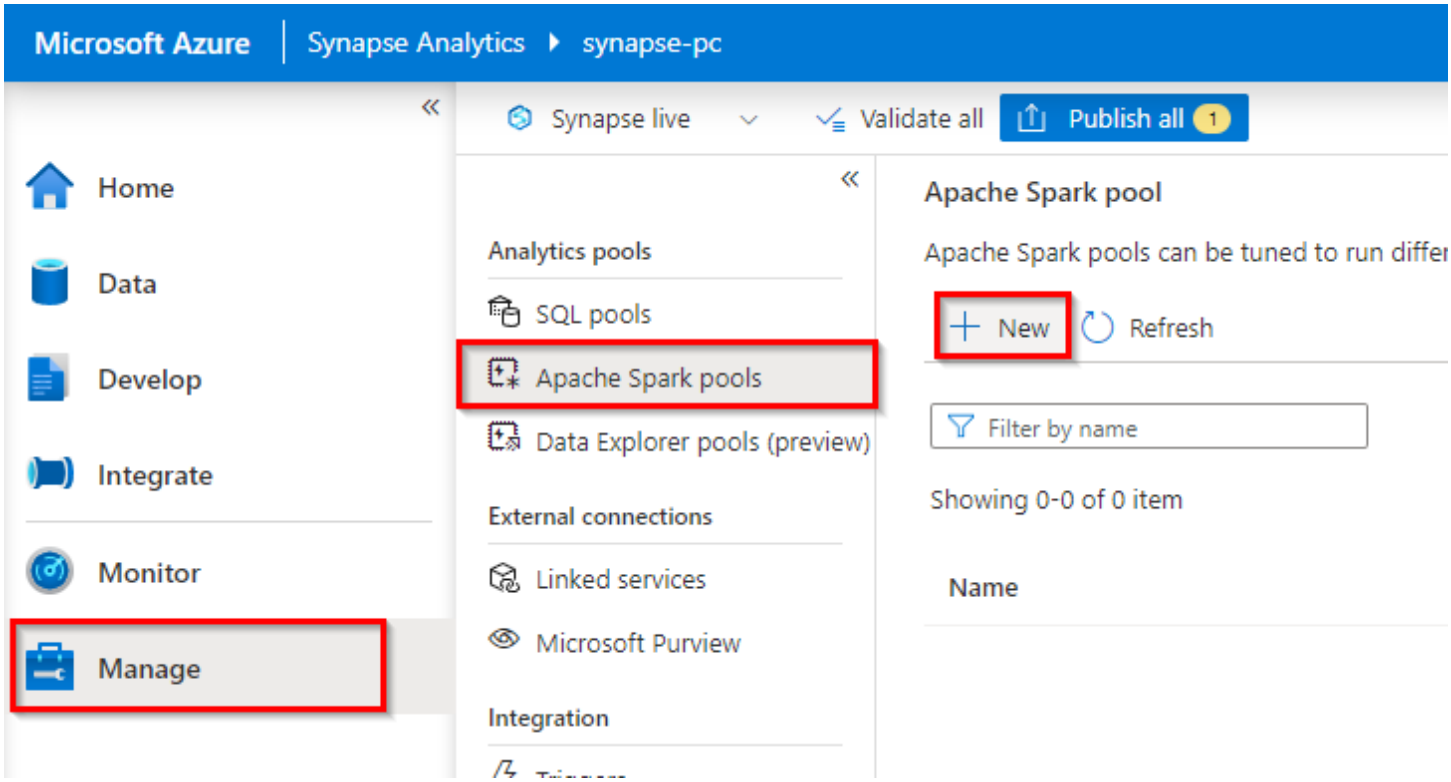


Task 6: Use a Spark pool to analyze data

In Azure Synapse Analytics, you can run Python (and other) code in a *Spark pool*; which uses a distributed data processing engine based on Apache Spark.

Create Spark Pool:

1. In Synapse Studio, under manage hub click on Apache spark pools and select + New.



2. In basic Tab:

   a. Apache spark pool name: sparkpool01
   b. Node size family: Memory Optimized
   c. Mode size: small (4 vCores / 32 GB)
   d. Autoscale: Enabled
   e. Number of nodes: 3 to 4

   Leave rest of the setting to default.

Explore and analyze the data using spark pool.

1. In Synapse Studio, if the **files** tab you opened earlier containing the **products.csv** file is no longer open, on the **Data** page, browse **product_data** folder. Then right-click **products.csv**, point to **New notebook**, and select **Load to DataFrame**.

2. In the **Notebook 1** pane that opens, in the **Attach to** list, select the **sparkxxxxxxx** Spark pool and ensure that the **Language** is set to **PySpark (Python)**.

3. Review the code in the first (and only) cell in the notebook, which should look like this:

```
%%pyspark
df = spark.read.load('abfss://files@asastoragexx.dfs.core.windows.net/product_data/products.csv',
format='csv'
## If header exists uncomment line below
##, header=True
)
display(df.limit(10))
```

4. Use the ▷ icon to the left of the code cell to run it, and wait for the results. The first time you run a cell in a notebook, the Spark pool is started - so it may take a minute or so to return any results.

5. Eventually, the results should appear below the cell, and they should be similar to this:

c0	***c1***	***c2***	***c3***
ProductID	ProductName	Category	ListPrice
771	Mountain-100 Silver, 38	Mountain Bikes	3399.9900
772	Mountain-100 Silver, 42	Mountain Bikes	3399.9900
...	...	...	...

6. Uncomment the *,header=True* line (because the products.csv file has the column headers in the first line), so your code looks like this:

```
%%pyspark
df =
spark.read.load('abfss://files@asastoragexx.dfs.core.windows.net/product_data/produc
ts.csv', format='csv'
## If header exists uncomment line below
, header=True
)
display(df.limit(10))
```

7. Rerun the cell and verify that the results look like this:

ProductID	**ProductName**	**Category**	**ListPrice**
771	Mountain-100 Silver, 38	Mountain Bikes	3399.9900
772	Mountain-100 Silver, 42	Mountain Bikes	3399.9900
...	...	...	...

8. Notice that running the cell again takes less time, because the Spark pool is already started.

9. Under the results, use the + **Code** icon to add a new code cell to the notebook.

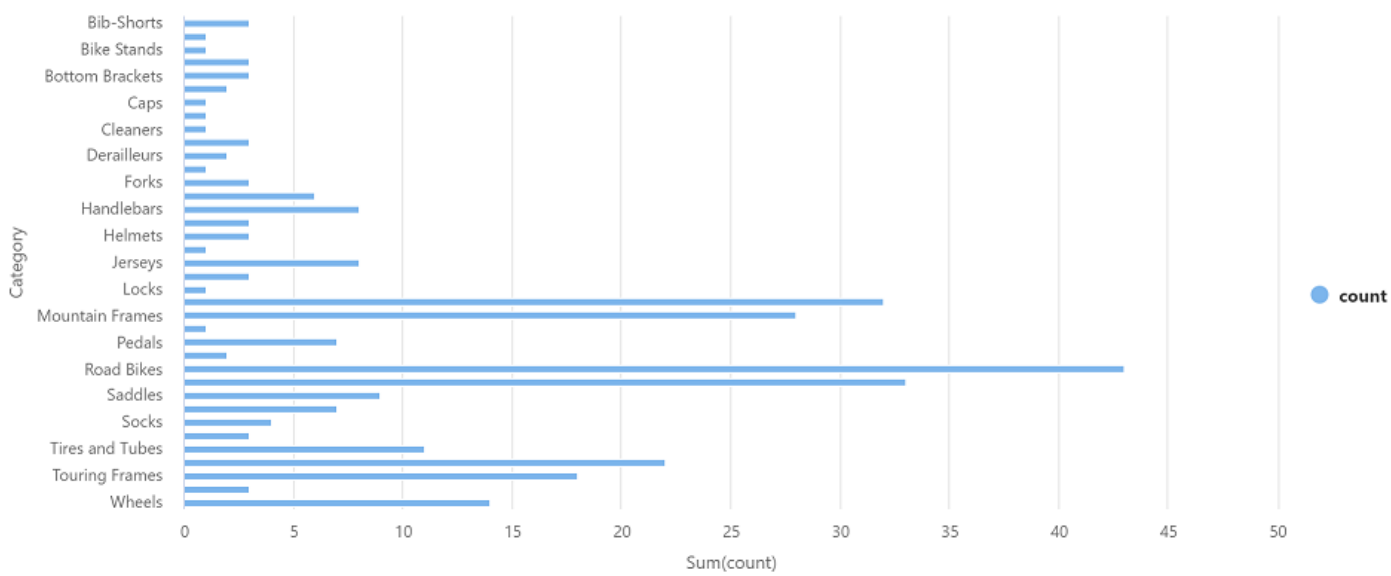
10. In the new empty code cell, add the following code:

    df_counts = df.groupby(df.Category).count()
    display(df_counts)

11. Run the new code cell by clicking its ▷ icon, and review the results, which should look similar to this:

Category	**count**
Headsets	3
Wheels	14
...	...

12. In the results output for the cell, select the **Chart** view. The resulting chart should resemble this:



13. If it is not already visible, show the **Properties** page by selecting the **Properties** button (which looks similar to 🗎*) on the right end of the toolbar. Then in the **Properties** pane, change the notebook name to **Explore products** and use the **Publish** button on the toolbar to save it.

14. Close the notebook pane and stop the Spark session when prompted. Then view the **Develop** page to verify that the notebook has been saved.