

Lab 4 Lake Database

BUAN 6390.001 – Analytics Practicum

Name: Swastik Bhatnagar
NetID: sxb220210

Connecting to PowerShell:

labclient.labondemand.com/LabClient/0e4d7bba-2211-4585-858e-22338b1798ee

azure portal login - Search x Home - Microsoft Azure x +

https://portal.azure.com/#home

Microsoft Azure

Search resources, services, and docs (5+)

Copilot

User1-4840616@L0DS...
L0DS-PROD-MCA-B00SPR0DM

Azure services

- Create a resource
- Quickstart Center
- Azure AI services
- Kubernetes services
- Virtual machines
- App Services
- Storage accounts
- SQL databases
- Azure Cosmos DB
- More services

Resources

Recent Favorite

Name	Type	Last Viewed
No resources have been viewed recently		

View all resources

Navigate

Switch to Bash Restart Manage files New session Editor Web preview Settings Help

```
Registering resource providers...
Microsoft.Synapse : Registering
Microsoft.Sql : Registering
Microsoft.Storage : Registering
Microsoft.Compute : Registering
Your randomly-generated suffix for Azure resources is zhdcw6m
Finding an available region. This may take several minutes...
Trying centralus
Using centralus
Creating dp203-zhdcw6m resource group in centralus ...
Creating synapsezhdcw6m Synapse Analytics workspace in dp203-zhdcw6m resource group...
(This may take some time!)
```

Lake Database

1 Hr 52 Min Remaining

100%

Instructions Resources Help

Note: If you have previously created a cloud shell that uses a bash environment, use the drop-down menu at the top left of the cloud shell pane to change it to **PowerShell**.

3. Note that you can resize the cloud shell by dragging the separator bar at the top of the pane, or by using the —, □, and X icons at the top right of the pane to minimize, maximize, and close the pane. For more information about using the Azure Cloud Shell, see the [Azure Cloud Shell documentation](#).

4. In the PowerShell pane, enter the following commands to clone this repo:

```
g/dp-203-azure-data-engineer dp-203
```

5. After the repo has been cloned, enter the following commands to change to the folder for this exercise and run the **setup.ps1** script it contains:

```
cd dp-203/Allfiles/Labs/04
./setup.ps1
```

6. If prompted, choose which subscription you want to use (this will only happen if you have access to multiple Azure subscriptions).

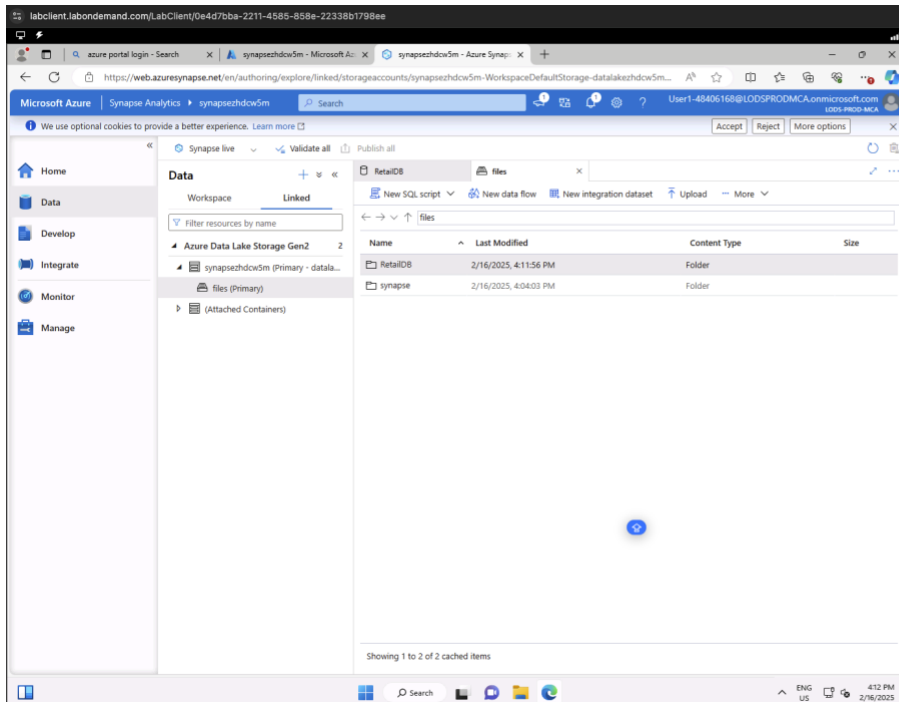
7. When prompted, enter a suitable password to be set for your Azure Synapse SQL pool.

Note: Be sure to remember this password!

8. Wait for the script to complete - this typically takes around 10 minutes, but in some cases may take longer. While you are waiting, review the [Lake database and Lake database templates](#) articles in the [Azure Synapse Analytics documentation](#).

End >

Creating a Lake Database:



Microsoft Azure | Synapse Analytics | synapsezhdcw5m | Search | User1-48406168@LODSPRODMDCA.onmicrosoft.com | LODS-PROD-MCA

Home | Data | Develop | Integrate | Monitor | Manage

Workspace: Linked

Filter resources by name

- Azure Data Lake Storage Gen2 2
- synapsezhdcw5m (Primary - data...
- files (Primary)
- (Attached Containers)

Files:

Name	Last Modified	Content Type	Size
RetailDB	2/16/2025, 4:11:56 PM	Folder	
synapse	2/16/2025, 4:04:03 PM	Folder	

Showing 1 to 2 of 2 cached items

Create a lake database

A lake database is a type of database that you can define in your workspace, and work with using the built-in serverless SQL pool.

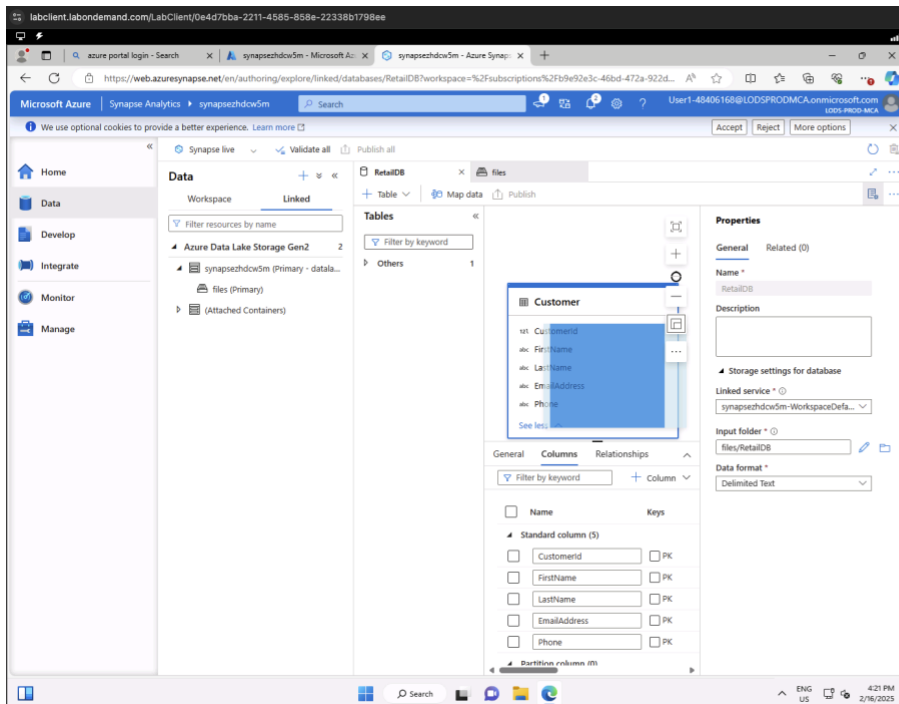
1. Select your Synapse workspace, and in its **Overview** page, in the **Open Synapse Studio** card, select **Open** to open Synapse Studio in a new browser tab; signing in if prompted.
2. On the left side of Synapse Studio, use the **»** icon to expand the menu - this reveals the different pages within Synapse Studio that you'll use to manage resources and perform data analytics tasks.
3. On the **Data** page, view the **Linked** tab and verify that your workspace includes a link to your Azure Data Lake Storage Gen2 storage account.
4. On the **Data** page, switch back to the **Workspace** tab and note that there are no databases in your workspace.
5. In the **+** menu, select **Lake database** to open a new tab in which you can design your database schema (accepting the database templates terms of use if prompted).
6. In the **Properties** pane for the new database, change the **Name** to **RetailDB** and verify that the **Input folder** property is automatically updated to **files/RetailDB**. Leave the **Data format** as **Delimited Text** (you could also use **Parquet** format, and you can override the file format for individual tables - we'll use comma-delimited data in this exercise.)
7. At the top of the **RetailDB** pane, select **Publish** to save the database so far.
8. In the **Data** pane on the left, view the **Linked** tab. Then expand **Azure Data Lake Storage Gen2** and the primary **datalake*xxxxxxx*** store for your **synapse*xxxxxxx*** workspace, and select the **files** file system; which currently contains a folder named **synapse**.
9. In the **files** tab that has opened, use the **+ New folder** button to create a new folder named **RetailDB** - this will be the input folder for the data files used by tables in your database.

Create a table

Now that you have created a lake database, you can define its schema by creating tables.

End >

Creating a Table:



Microsoft Azure | Synapse Analytics | synapsezhdcw5m | Search | User1-48406168@LODSPRODMDCA.onmicrosoft.com | LODS-PROD-MCA

Home | Data | Develop | Integrate | Monitor | Manage

Workspace: Linked

Filter resources by name

- Azure Data Lake Storage Gen2 2
- synapsezhdcw5m (Primary - data...
- files (Primary)
- (Attached Containers)

Tables:

Filter by keyword

Others 1

Customer

Columns: CustomerId, FirstName, LastName, EmailAddress, Phone

Properties:

General

Name: Customer

Description: Customer table

Storage settings for database

Linked service: synapsezhdcw5m-WorkspaceDefault

Input folder: files/RetailDB

Data format: Delimited Text

Create a table

Now that you have created a lake database, you can define its schema by creating tables.

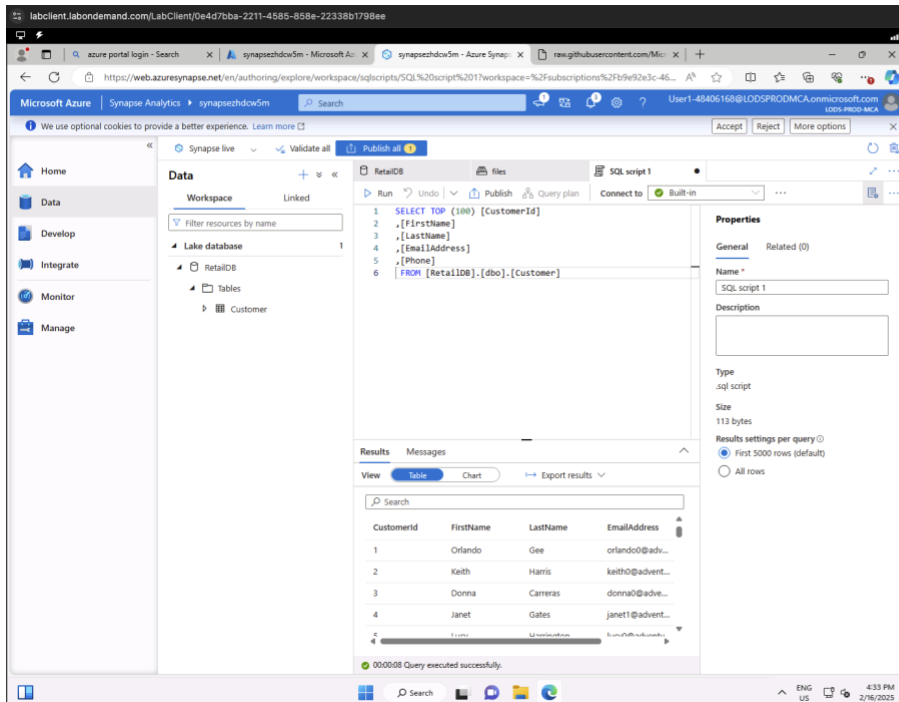
Name	Keys	Description	Nullability	Data type	Format / Length
CustomerId	PK	Unique customer ID	<input type="checkbox"/>	long	
FirstName	PK	Customer first name	<input type="checkbox"/>	string	256
LastName	PK	Customer last name	<input type="checkbox"/>	string	256
EmailAddress	PK	Customer email	<input type="checkbox"/>	string	256
Phone	PK	Customer phone	<input type="checkbox"/>	string	256

1. In the main pane, switch back to the **files** tab, which contains the file system with the **RetailDB** folder. Then open the **RetailDB** folder and create a new folder named **Customer** in it. This is where the **Customer** table will get its data.
2. Open the new **Customer** folder, which should be empty.
3. Download the **customer.csv** data file from <https://raw.githubusercontent.com/MicrosoftLearning/dp-203-azure-data-engineer/master/Files/lab04/data/customer.csv> and save it in a folder on your local computer (it doesn't matter where). Then in the **Customer** folder in Synapse Explorer, use the **Upload** button to upload the **customer.csv** file to the **RetailDB/Customer** folder in your data lake.

Note: In a real production scenario, you would probably create a pipeline to ingest data into the folder for the table data. We're uploading it directly in the Synapse Studio user interface in this exercise for expediency.

End >

Loading data into the table's storage path:



The screenshot shows the Microsoft Azure Synapse Analytics interface. The left sidebar contains navigation options: Home, Data, Develop, Integrate, Monitor, and Manage. The main pane is divided into three sections: Data, SQL script 1, and Properties. The Data section shows a tree view with 'Lake database' and 'RetailDB'. The SQL script 1 section contains a query:

```
SELECT TOP (100) [CustomerId]
, [FirstName]
, [LastName]
, [EmailAddress]
, [Phone]
FROM [RetailDB].[dbo].[Customer]
```

 The Properties section shows the script's name, description, type (SQL script), and size (113 bytes). The Results section displays a table with 4 rows of data:

CustomerId	FirstName	LastName	EmailAddress
1	Orlando	Gee	orlando0@advent...
2	Keith	Harris	keith0@advent...
3	Donna	Carreras	donna0@advent...
4	Janet	Gates	janet1@advent...

On the right side of the interface, there is a 'Lake Database' sidebar with instructions for loading data into the table's storage path:

1. In the main pane, switch back to the **files** tab, which contains the file system with the **RetailDB** folder. Then open the **RetailDB** folder and create a new folder named **Customer** in it. This is where the **Customer** table will get its data.
2. Open the new **Customer** folder, which should be empty.
3. Download the **customer.csv** data file from <https://raw.githubusercontent.com/MicrosoftLearning/tp-203-azure-data-engineer/master/Allfiles/04/data/customer.csv> and save it in a folder on your local computer (it doesn't matter where). Then in the **Customer** folder in Synapse Explorer, use the **Upload** button to upload the **customer.csv** file to the **RetailDB/Customer** folder in your data lake.

Note: In a real production scenario, you would probably create a pipeline to ingest data into the folder for the table data. We're uploading it directly in the Synapse Studio user interface in this exercise for expediency.

4. In the **Data** pane on the left, switch back to the **Workspace** tab so you can see the **RetailDB** lake database. Then expand it and refresh its **Tables** folder to see the newly created **Customer** table.
5. Close the **SQL script** tab, discarding your changes.

Create a table from a database template

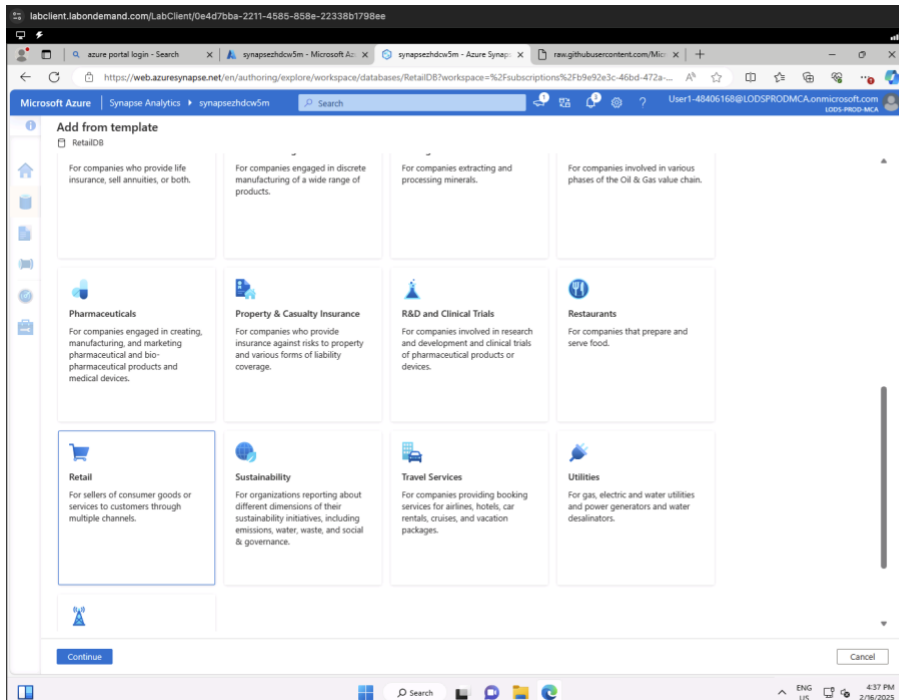
If you've seen, you can create the tables you need in your lake database from scratch. However, Azure Synapse Analytics also provides numerous database templates based on common database workloads and entities that you can use as a starting point for your database schema.

Define the table schema

1. In the main pane, switch back to the **RetailDB** pane, which contains your database schema (currently containing only the **Customer** table).
2. In the **+ Table** menu, select **From template**. Then in the **Add from template** page, select **Retail** and click **Continue**.

End >

Creating a table from a database template:



The screenshot shows the Microsoft Azure Synapse Analytics interface. The left sidebar contains navigation options: Home, Data, Develop, Integrate, Monitor, and Manage. The main pane is divided into three sections: Data, SQL script 1, and Properties. The Data section shows a tree view with 'Lake database' and 'RetailDB'. The SQL script 1 section contains a query:

```
SELECT TOP (100) [CustomerId]
, [FirstName]
, [LastName]
, [EmailAddress]
, [Phone]
FROM [RetailDB].[dbo].[Customer]
```

 The Properties section shows the script's name, description, type (SQL script), and size (113 bytes). The Results section displays a table with 4 rows of data:

CustomerId	FirstName	LastName	EmailAddress
1	Orlando	Gee	orlando0@advent...
2	Keith	Harris	keith0@advent...
3	Donna	Carreras	donna0@advent...
4	Janet	Gates	janet1@advent...

On the right side of the interface, there is a 'Lake Database' sidebar with instructions for creating a table from a database template:

1. In the main pane, switch back to the **RetailDB** pane, which contains your database schema (currently containing only the **Customer** table).
2. In the **+ Table** menu, select **From template**. Then in the **Add from template** page, select **Retail** and click **Continue**.
3. In the **Add from template (Retail)** page, wait for the table list to populate, and then expand **Product** and select **RetailProduct**. Then click **Add**. This adds a new table based on the **RetailProduct** template to your database.
4. In the **RetailDB** pane, select the new **RetailProduct** table. Then, in the pane beneath the design canvas, on the **General** tab, change the name to **Product** and verify that the storage settings for the table specify the input folder **files/RetailDB/Product**.
5. On the **Columns** tab for the **Product** table, note that the table already includes a large number of columns inherited from the template. There are more columns than required for this table, so you'll need to remove some.
6. Select the checkbox next to **Name** to select all of the columns, and then **unselect** the following columns (which you need to retain):
 - ProductId
 - ProductName
 - IntroductionDate
 - ActualAbandonmentDate
 - ProductGrossWeight
 - ItemSku
7. On the toolbar in the **Columns** pane, select **Delete** to remove the selected columns. This should leave you with the following columns:

Define the table schema

Name	Keys	Description	Nullability	Data type
ProductId	PK	The unique identifier of a Product.	<input type="checkbox"/>	long
ProductName	PK	The name of the Product.	<input type="checkbox"/>	string
IntroductionDate	PK	The date that the Product was introduced.	<input type="checkbox"/>	date

End >

Home

Data

Develop

Integrate

Monitor

Manage

Workspace

Linked

Filter resources by name

Lake database

RetailDB

Tables

Customer

Product

Tables

Filter by keyword

Product

ProductID

ProductName

IntroductionDate

ActualAbandonmentDate

ProductGrossWeight

ItemSku

ListPrice

Properties

General

Related (0)

Name *

RetailDB

Description

Lake database template

Retail 1.3.0

Storage settings for database

Linked service *

synapsezhdcw5m-WorkspaceDefa...

Input folder *

files/RetailDB

Data format *

Delimited Text

General

Columns

Relationships

Filter by keyword

Column

Name

Keys

Standard column (7)

ProductID

ProductName

IntroductionDate

ActualAbandonmentDate

ProductGrossWeight

ItemSku

ListPrice

Lake Database

1 Hr 4 Min Remaining

Instructions

Resources

Help

100%

ColumnName	PK	Description	DataType
ProductName	<input type="checkbox"/>	The name of the Product...	string
IntroductionDate	<input type="checkbox"/>	The date that the Product was introduced for sale.	date
ActualAbandonmentDate	<input type="checkbox"/>	The actual date that the marketing of the product was discontinued...	date
ProductGrossWeight	<input type="checkbox"/>	The gross product weight.	decr
ItemSku	<input type="checkbox"/>	The Stock Keeping Unit identifier...	string
ListPrice	<input type="checkbox"/>	The product price.	decr

9. When you've modified the columns as shown above, publish the database again to save the changes.

10. In the **Data** pane on the left, switch back to the **Workspace** tab so you can see the **RetailDB** lake database. Then use the ... menu for its **Tables** folder to refresh the view and see the newly created **Product** table.

Load data into the table's storage path

1. In the main pane, switch back to the **files** tab, which contains the file system, and navigate to the **files/RetailDB** folder, which currently contains the **Customer** folder for the table you created previously.

2. In the **RetailDB** folder, create a new folder named **Product**. This is where the **Product** table will get its data.

3. Open the new **Product** folder, which should be empty.

4. Download the **product.csv** data file from <https://raw.githubusercontent.com/MicrosoftLearning/dp-203-azure-data-engineer/master/AIFiles/labs/04/data/product.csv> and save it in a folder

Loading data into table's storage path:

Home

Data

Develop

Integrate

Monitor

Manage

Workspace

Linked

Filter resources by name

Lake database

RetailDB

Tables

Customer

Product

Tables

Filter by keyword

Product

ProductID

ProductName

IntroductionDate

ActualAbandonmentDate

ProductGrossWeight

ItemSku

ListPrice

Properties

General

Related (0)

Name *

SQL script 1

Description

Type

sql script

Size

167 Bytes

Results settings per query (0)

☒ First 5000 rows (default)

☐ All rows

General

Columns

Relationships

Filter by keyword

Column

Name

Keys

Standard column (7)

ProductID

ProductName

IntroductionDate

ActualAbandonmentDate

ProductGrossWeight

ItemSku

ListPrice

Lake Database

1 Hour Remaining

Instructions

Resources

Help

100%

table.

Load data into the table's storage path

1. In the main pane, switch back to the **files** tab, which contains the file system, and navigate to the **files/RetailDB** folder, which currently contains the **Customer** folder for the table you created previously.

2. In the **RetailDB** folder, create a new folder named **Product**. This is where the **Product** table will get its data.

3. Open the new **Product** folder, which should be empty.

4. Download the **product.csv** data file from <https://raw.githubusercontent.com/MicrosoftLearning/dp-203-azure-data-engineer/master/AIFiles/labs/04/data/product.csv> and save it in a folder on your local computer (it doesn't matter where). Then in the **Product** folder in Synapse Explorer, use the **Upload** button to upload the **product.csv** file to the **RetailDB/Product** folder in your data lake.

5. In the **Data** pane on the left, on the **Workspace** tab, in the ... menu for the **Product** table, select **New SQL script > Select TOP 100 rows**. Then, in the new **SQL script 1** pane that has opened, ensure that the **Built-in** SQL pool is connected, and use the **Run** button to run the SQL code. The results should include first 100 rows from the **Product** table, based on the data stored in the underlying folder in the data lake.

6. Close the **SQL script 1** tab, discarding your changes.

Create a table from existing data

So far, you've created tables and then populated them with data. In some cases, you may already have data in a data lake from which you want to derive a table.

Upload data

1. In the main pane, switch back to the **files** tab, which contains the file system, and navigate to the **files/RetailDB** folder, which currently contains the **Customer** and **Product** folders for the tables you created previously.

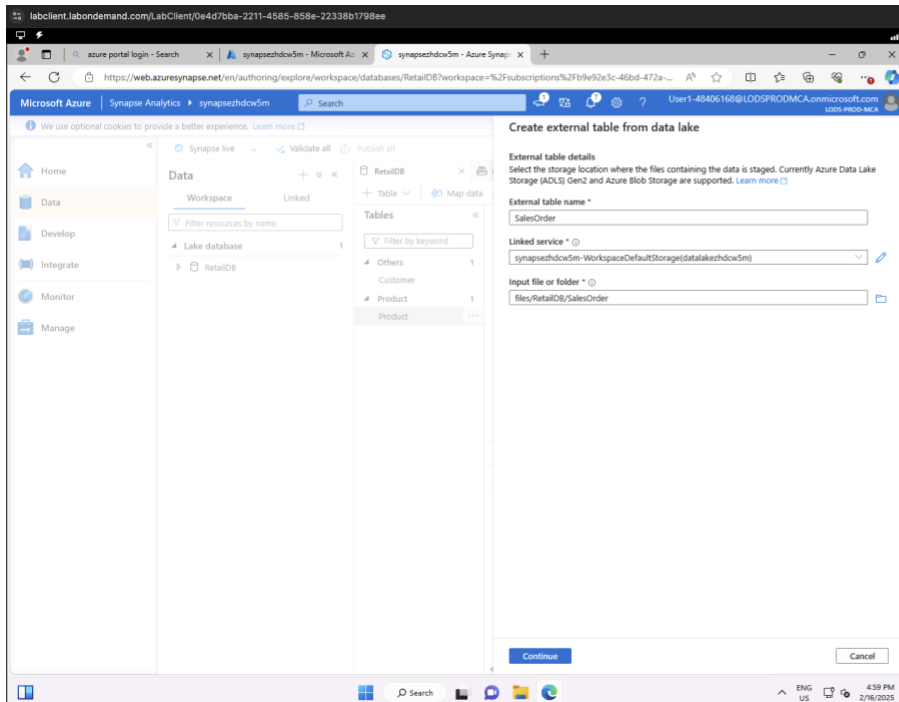
2. In the **RetailDB** folder, create a new folder named **SalesOrder**.

3. Open the new **SalesOrder** folder, which should be empty.

4. Download the **salesorder.csv** data file from <https://raw.githubusercontent.com/MicrosoftLearning/dp-203-azure-data-engineer/master/AIFiles/labs/04/data/salesorder.csv> and save it in a folder on your local computer (it doesn't matter where). Then in the **SalesOrder** folder in Synapse Explorer, use the **Upload** button to upload the **salesorder.csv** file to the **RetailDB/SalesOrder** folder in your data lake.

Create a table

Creating a table from existing data:



Create external table from data lake

External table details
Select the storage location where the files containing the data is staged. Currently Azure Data Lake Storage (ADLS) Gen2 and Azure Blob Storage are supported. [Learn more](#)

External table name *
SalesOrder

Linked service *
synapsehdcw5m-WorkspaceDefaultStorage(datalakehdcw5m)

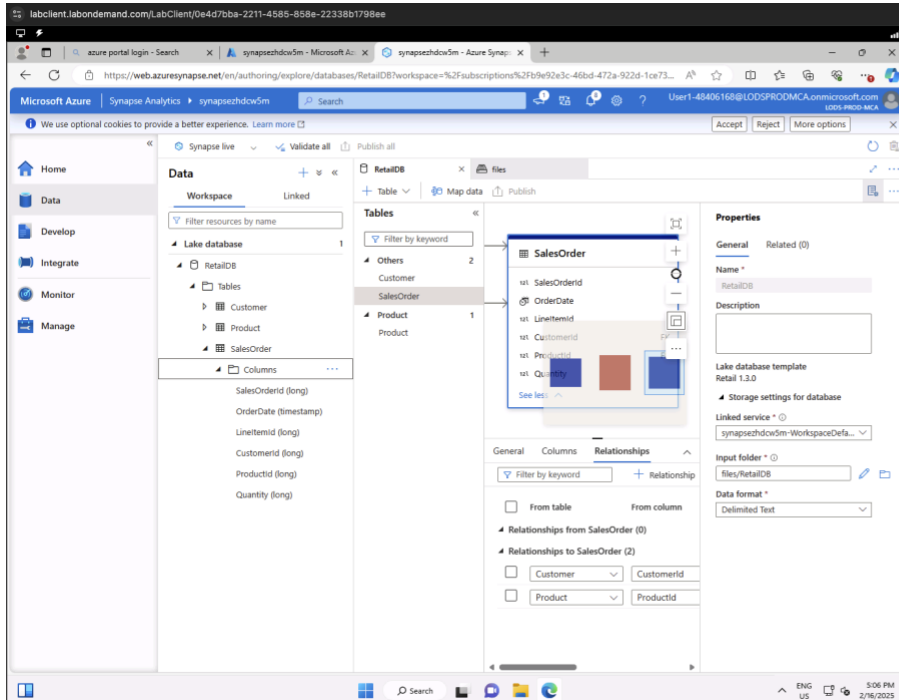
Input file or folder *
files/RetailDB/SalesOrder

Create a table

- In the main pane, switch back to the **RetailDB** pane, which contains your database schema (currently containing the **Customer** and **Product** tables).
- In the + **Table** menu, select **From data lake**. Then in the **Create external table from data lake** pane, specify the following options:
 - External table name:** SalesOrder
 - Linked service:** Select **synapse*xxxxxxxx***.
WorkspaceDefaultStorage(datalake*xxxxxxxx*)
 - Input file or folder:** files/RetailDB/SalesOrder
- Continue to the next page and then create the table with the following options:
 - File type:** CSV
 - Field terminator:** Default (comma ,)
 - First row:** Leave *infer column names* unselected.
 - String delimiter:** Default (Empty string)
 - Use default type:** Default type (true,false)
 - Max string length:** 4000
- When the table has been created, note that it includes columns named **C1**, **C2**, and so on and that the data types have been inferred from the data in the folder. Modify the column definitions as follows:

Name	Keys	Description	Nullability	Data type	Form / Leng
SalesOrderId	PK	The unique identifier of an order.	<input type="checkbox"/>	long	
OrderDate	PK	The date of the order.	<input type="checkbox"/>	timestamp	YYYY MM-
		The ID of			

End >



Work with lake database tables

Now that you have some tables in your database, you can use them to work with the underlying data.

Query tables using SQL

- In Synapse Studio, select the **Develop** page.
- In the **Develop** pane, in the + menu, select **SQL script**.
- In the new **SQL script** pane, ensure the script is connected to the **Built-in SQL pool** and in the **User database** list, select **RetailDB**.
- Enter the following SQL code:

```
sql
SELECT o.SalesOrderId, c.EmailAddress, p.ProductName
FROM SalesOrder AS o
JOIN Customer AS c ON o.CustomerId = c.CustomerId
JOIN Product AS p ON o.ProductId = p.ProductId
```

- Use the **Run** button to run the SQL code.
- The results show order details with customer and product information.
- Close the **SQL script** pane, discarding your changes.

Insert data using Spark

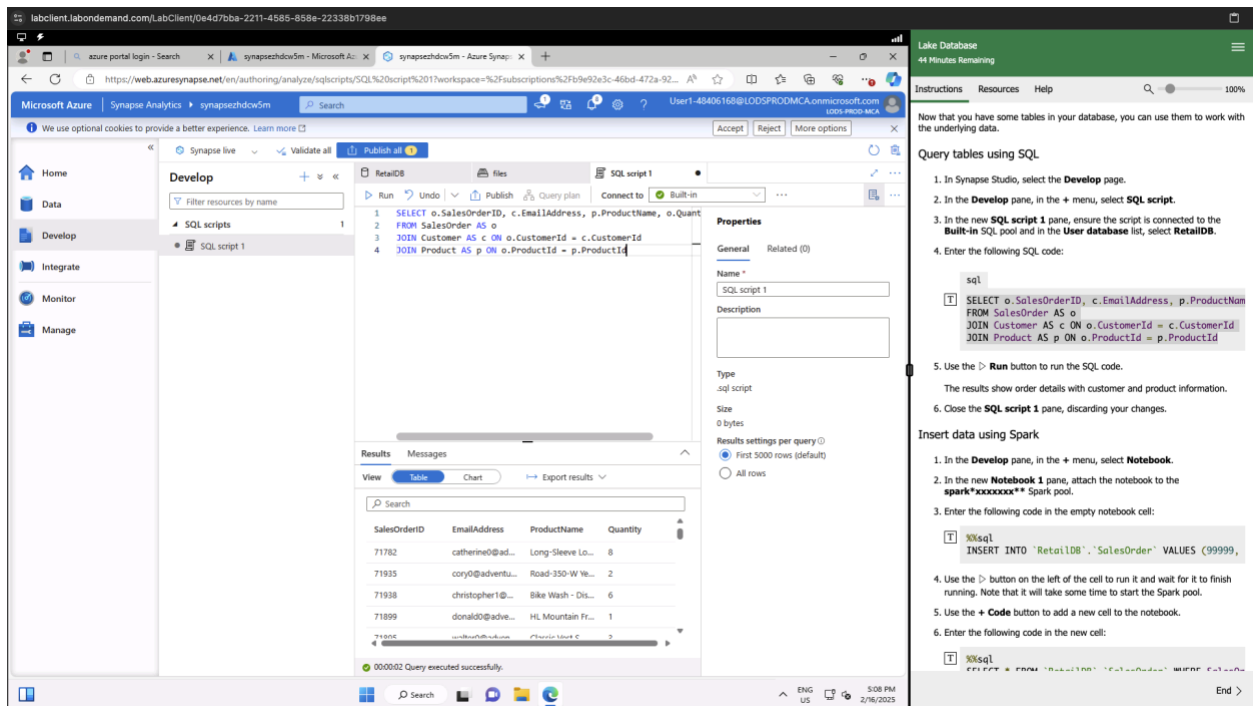
- In the **Develop** pane, in the + menu, select **Notebook**.
- In the new **Notebook** pane, attach the notebook to the **spark*xxxxxxxx*** Spark pool.
- Enter the following code in the empty notebook cell:

```
%sql
INSERT INTO 'RetailDB'.SalesOrder VALUES ('99999',
```

- Use the **Run** button on the left of the cell to run it and wait for it to finish running. Note that it will take some time to start the Spark pool.
- Use the + **Code** button to add a new cell to the notebook.
- Enter the following code in the new cell:

End >

Now I have some tables in my database so I can use them to work with the underlying data:



Microsoft Azure | Synapse Analytics | synapsezhdcw5m | Search | User1-48406168@LODSPRODCA.onmicrosoft.com | 100% PROO-MCA

Develop

SQL scripts

SQL script 1

```
1 SELECT o.SalesOrderID, c.EmailAddress, p.ProductName, o.Quantity
2 FROM SalesOrder AS o
3 JOIN Customer AS c ON o.CustomerId = c.CustomerId
4 JOIN Product AS p ON o.ProductId = p.ProductId
```

Properties

General

Name: SQL script 1

Description:

Type: sql script

Size: 0 bytes

Results settings per query: First 5000 rows (default)

Results

Table

SalesOrderID	EmailAddress	ProductName	Quantity
71782	catherine0@ad...	Long Sleeve Lo...	8
71935	cory0@adventu...	Road-350-W Ye...	2
71938	christopher1@...	Bike Wash - Dis...	6
71899	dona10@adve...	HL Mountain Fr...	1
71902	don10@adve...	Phantom Vant C...	2

300082 Query executed successfully.

Query tables using SQL

1. In Synapse Studio, select the **Develop** page.
2. In the **Develop** pane, in the + menu, select **SQL script**.
3. In the new **SQL script** pane, ensure the script is connected to the **Built-in** SQL pool and in the **User database** list, select **RetailDB**.
4. Enter the following SQL code:

```
1 SELECT o.SalesOrderID, c.EmailAddress, p.ProductName
FROM SalesOrder AS o
JOIN Customer AS c ON o.CustomerId = c.CustomerId
JOIN Product AS p ON o.ProductId = p.ProductId
```

5. Use the **Run** button to run the SQL code.
- The results show order details with customer and product information.
6. Close the **SQL script** pane, discarding your changes.

Insert data using Spark

1. In the **Develop** pane, in the + menu, select **Notebook**.
2. In the new **Notebook** pane, attach the notebook to the **sparkxxxxxxx** Spark pool.
3. Enter the following code in the empty notebook cell:

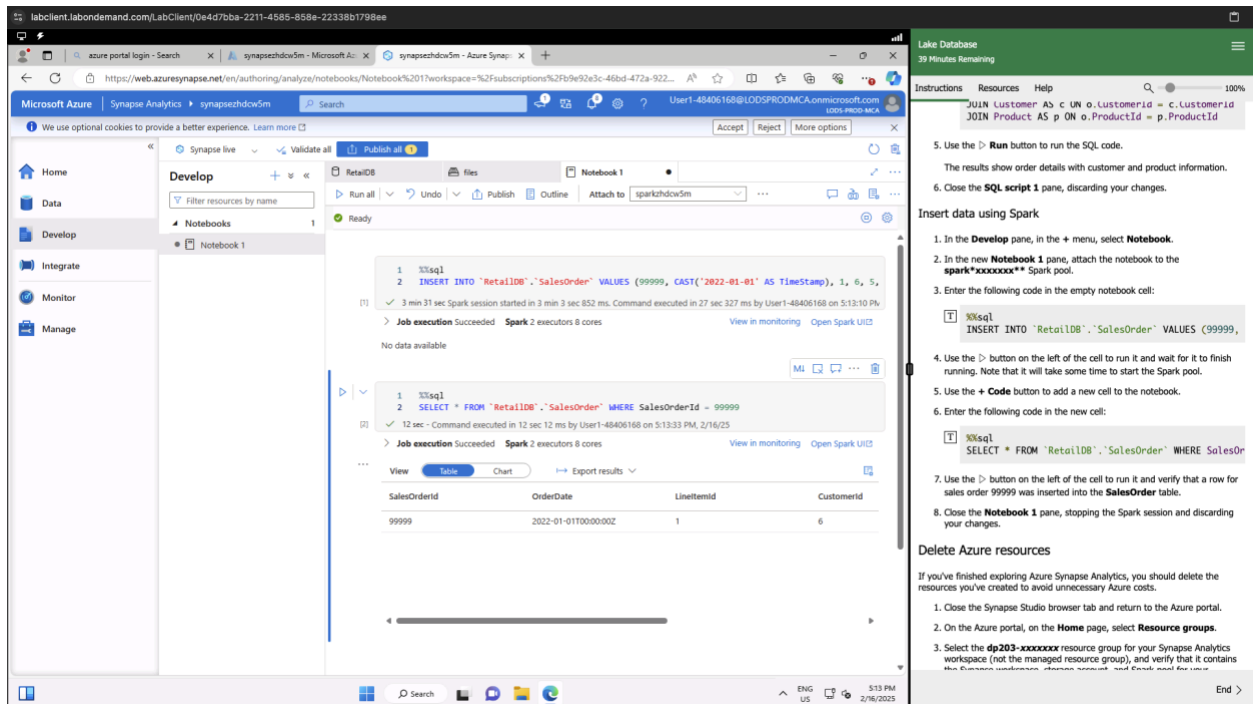
```
1 %sql
INSERT INTO 'RetailDB'.'SalesOrder' VALUES (99999, '2022-01-01' AS Timestamp, 1, 6, 5,
```

4. Use the **Run** button on the left of the cell to run it and wait for it to finish running. Note that it will take some time to start the Spark pool.
5. Use the + **Code** button to add a new cell to the notebook.
6. Enter the following code in the new cell:

```
1 %sql
SELECT * FROM 'RetailDB'.'SalesOrder' WHERE SalesOrderID = 99999
```

End >

Inserting Data using Spark:



Microsoft Azure | Synapse Analytics | synapsezhdcw5m | Search | User1-48406168@LODSPRODCA.onmicrosoft.com | 100% PROO-MCA

Develop

Notebooks

Notebook 1

```
1 %sql
2 INSERT INTO 'RetailDB'.'SalesOrder' VALUES (99999, CAST('2022-01-01' AS Timestamp), 1, 6, 5,
```

Job execution Succeeded Spark 2 executors 8 cores

No data available

```
1 %sql
2 SELECT * FROM 'RetailDB'.'SalesOrder' WHERE SalesOrderID = 99999
```

Job execution Succeeded Spark 2 executors 8 cores

Results

Table

SalesOrderID	OrderDate	LineItemID	CustomerID
99999	2022-01-01T00:00:00Z	1	6

5:13 PM 2/16/2025

Query tables using SQL

1. In Synapse Studio, select the **Develop** page.
2. In the **Develop** pane, in the + menu, select **SQL script**.
3. In the new **SQL script** pane, ensure the script is connected to the **Built-in** SQL pool and in the **User database** list, select **RetailDB**.
4. Enter the following SQL code:

```
1 SELECT o.SalesOrderID, c.EmailAddress, p.ProductName
FROM SalesOrder AS o
JOIN Customer AS c ON o.CustomerId = c.CustomerId
JOIN Product AS p ON o.ProductId = p.ProductId
```

5. Use the **Run** button to run the SQL code.
- The results show order details with customer and product information.
6. Close the **SQL script** pane, discarding your changes.

Insert data using Spark

1. In the **Develop** pane, in the + menu, select **Notebook**.
2. In the new **Notebook** pane, attach the notebook to the **sparkxxxxxxx** Spark pool.
3. Enter the following code in the empty notebook cell:

```
1 %sql
INSERT INTO 'RetailDB'.'SalesOrder' VALUES (99999,
```

4. Use the **Run** button on the left of the cell to run it and wait for it to finish running. Note that it will take some time to start the Spark pool.
5. Use the + **Code** button to add a new cell to the notebook.
6. Enter the following code in the new cell:

```
1 %sql
SELECT * FROM 'RetailDB'.'SalesOrder' WHERE SalesOrderID = 99999
```

7. Use the **Run** button on the left of the cell to run it and verify that a row for sales order 99999 was inserted into the **SalesOrder** table.
8. Close the **Notebook** pane, stopping the Spark session and discarding your changes.

Delete Azure resources

If you've finished exploring Azure Synapse Analytics, you should delete the resources you've created to avoid unnecessary Azure costs.

1. Close the Synapse Studio browser tab and return to the Azure portal.
2. On the Azure portal, on the **Home** page, select **Resource groups**.
3. Select the **dp203-xxxxxxx** resource group for your Synapse Analytics workspace (not the managed resource group), and verify that it contains the **Synapse workspace**, **storage account**, and **Event Hubs** resources.

End >

Now Deleting Azure resources:

The screenshot displays the Microsoft Azure portal interface. The main content area shows the 'Resources' tab for the resource group 'dp203-zhdcw5m'. A table lists the resources within this group:

Name	Type	Location
datalakezhdcw5m	Storage account	Central US
sparkzhdcw5m (synapsezhdcw5m)	Apache Spark pool	Central US
synapsezhdcw5m	Synapse workspace	Central US

A notification banner at the top right indicates 'Deleting resource group dp203-zhdcw5m' with a status of 'Running'.

On the right side, a 'Lake Database' sidebar is visible, showing a list of instructions for deleting Azure resources. The instructions include:

- Close the Synapse Studio browser tab and return to the Azure portal.
- On the Azure portal, on the **Home** page, select **Resource groups**.
- Select the **dp203-xxxxxxx** resource group for your Synapse Analytics workspace (not the managed resource group), and verify that it contains the Synapse workspace, storage account, and Spark pool for your workspace.
- At the top of the **Overview** page for your resource group, select **Delete resource group**.
- Enter the **dp203-xxxxxxx** resource group name to confirm you want to delete it, and select **Delete**.

After a few minutes, your Azure Synapse workspace resource group and the managed workspace resource group associated with it will be deleted.

The bottom of the screen shows a terminal window with the following output:

```
Script completed at 02/17/2025 00:04:40
PS /home/user1-48406168/dp-203/Allfiles/labs/04>
```

Conclusion: In this lab, I learned how to create and manage a Lake Database in Azure. I created tables, loaded data into storage paths, and worked with database templates. Additionally, I used Spark to insert data and explored different ways to interact with the underlying data. Finally, I practiced resource management by deleting Azure resources. This lab enhanced my understanding of data storage and processing in a cloud environment.