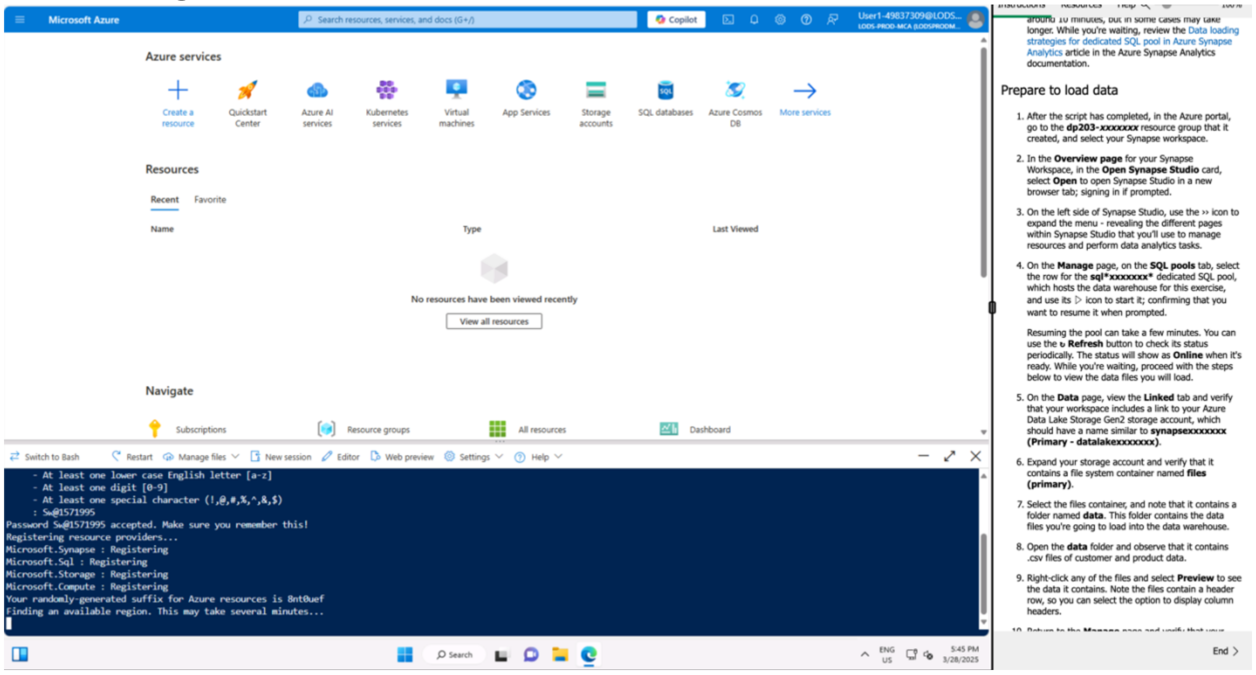
Lab 9 Load Data Warehouse
BUAN 6390.001 – Analytics Practicum

Name: Swastik Bhatnagar
Net ID: Sxb220210

Connecting to PowerShell:



```
    - At least one lower case English letter [a-z]
    - At least one digit [0-9]
    - At least one special character (!,@,#,%,^,&,$)
    : Sw@1571995
Password Sw@1571995 accepted. Make sure you remember this!
Registering resource providers...
Microsoft.Synapse : Registering
Microsoft.Sql : Registering
Microsoft.Storage : Registering
Microsoft.Compute : Registering
Your randomly-generated suffix for Azure resources is 8nt0uef
Finding an available region. This may take several minutes...
```

longer. While you're waiting, review the Data loading strategies for dedicated SQL pool in Azure Synapse Analytics article in the Azure Synapse Analytics documentation.

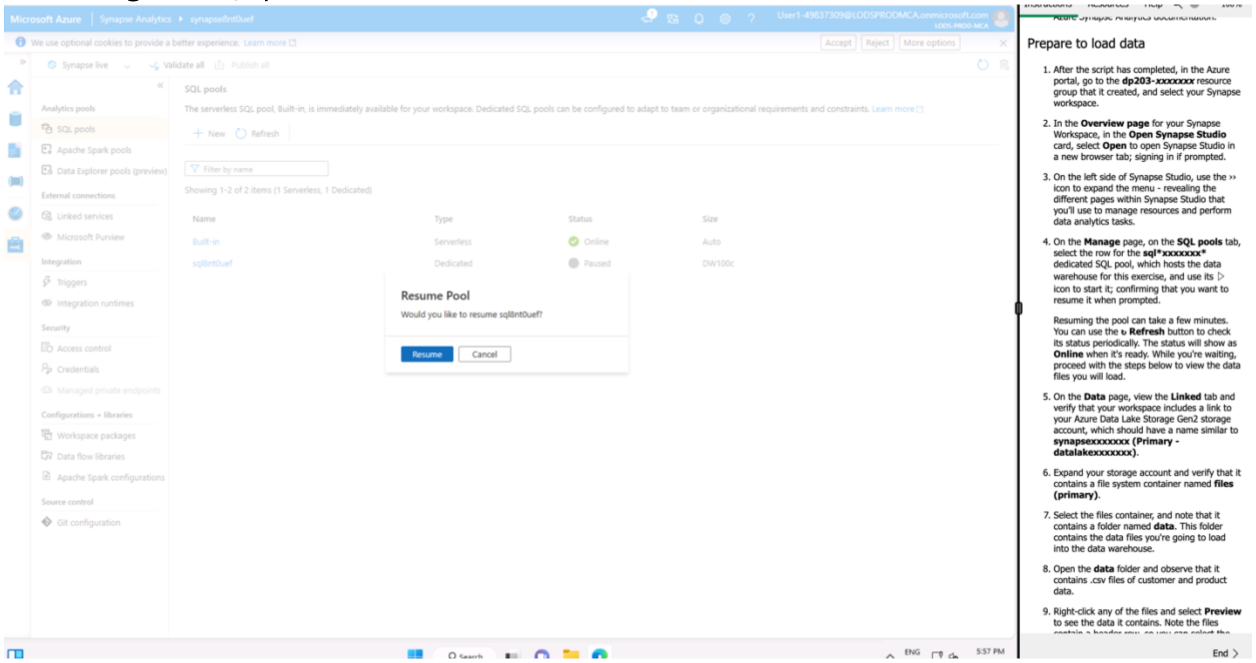
Prepare to load data

1. After the script has completed, in the Azure portal, go to the **dp203-*xxxxxxx*** resource group that it created, and select your Synapse workspace.
2. In the **Overview page** for your Synapse Workspace, in the **Open Synapse Studio** card, select **Open** to open Synapse Studio in a new browser tab; signing in if prompted.
3. On the left side of Synapse Studio, use the ›› icon to expand the menu - revealing the different pages within Synapse Studio that you'll use to manage resources and perform data analytics tasks.
4. On the **Manage** page, on the **SQL pools** tab, select the row for the **sql*xxxxxxx*** dedicated SQL pool, which hosts the data warehouse for this exercise, and use its ▷ icon to start it; confirming that you want to resume it when prompted.

   Resuming the pool can take a few minutes. You can use the ↻ **Refresh** button to check its status periodically. The status will show as **Online** when it's ready. While you're waiting, proceed with the steps below to view the data files you will load.
5. 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, which should have a name similar to **synapsexxxxxxx (Primary - datalakexxxxxxx)**.
6. Expand your storage account and verify that it contains a file system container named **files (primary)**.
7. Select the files container, and note that it contains a folder named **data**. This folder contains the data files you're going to load into the data warehouse.
8. Open the **data** folder and observe that it contains .csv files of customer and product data.
9. Right-click any of the files and select **Preview** to see the data it contains. Note the files contain a header row, so you can select the option to display column headers.

End >

Resuming the SQL pool:



SQL pools

The serverless SQL pool, Built-in, is immediately available for your workspace. Dedicated SQL pools can be configured to adapt to team or organizational requirements and constraints. Learn more

Showing 1-2 of 2 items (1 Serverless, 1 Dedicated)

Name	Type	Status	Size
Built-in	Serverless	✓ Online	Auto
sql8nt0uef	Dedicated	● Paused	DW100c

Resume Pool

Would you like to resume sql8nt0uef?

Prepare to load data

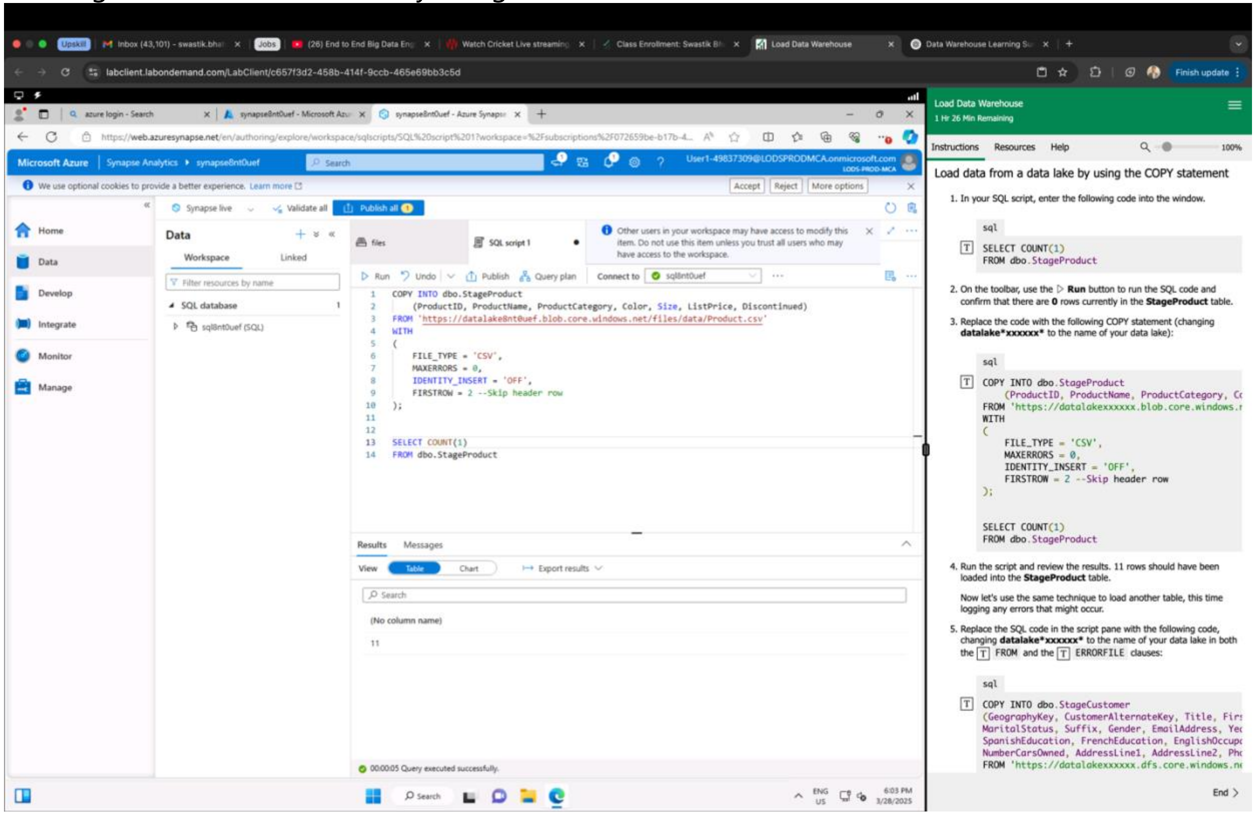
1. After the script has completed, in the Azure portal, go to the **dp203-*xxxxxxx*** resource group that it created, and select your Synapse workspace.
2. In the **Overview page** for your Synapse Workspace, in the **Open Synapse Studio** card, select **Open** to open Synapse Studio in a new browser tab; signing in if prompted.
3. On the left side of Synapse Studio, use the ›› icon to expand the menu - revealing the different pages within Synapse Studio that you'll use to manage resources and perform data analytics tasks.
4. On the **Manage** page, on the **SQL pools** tab, select the row for the **sql*xxxxxxx*** dedicated SQL pool, which hosts the data warehouse for this exercise, and use its ▷ icon to start it; confirming that you want to resume it when prompted.

   Resuming the pool can take a few minutes. You can use the ↻ **Refresh** button to check its status periodically. The status will show as **Online** when it's ready. While you're waiting, proceed with the steps below to view the data files you will load.
5. 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, which should have a name similar to **synapsexxxxxxx (Primary - datalakexxxxxxx)**.
6. Expand your storage account and verify that it contains a file system container named **files (primary)**.
7. Select the files container, and note that it contains a folder named **data**. This folder contains the data files you're going to load into the data warehouse.
8. Open the **data** folder and observe that it contains .csv files of customer and product data.
9. Right-click any of the files and select **Preview** to see the data it contains. Note the files

End >

Loading the Data warehouse table:

Loading data from a data lake by using the COPY statement:



```sql
COPY INTO dbo.StageProduct
    (ProductID, ProductName, ProductCategory, Color, Size, ListPrice, Discontinued)
FROM 'https://datalake8nt0uef.blob.core.windows.net/files/data/Product.csv'
WITH
(
    FILE_TYPE = 'CSV',
    MAXERRORS = 0,
    IDENTITY_INSERT = 'OFF',
    FIRSTROW = 2 --Skip header row
);


SELECT COUNT(1)
FROM dbo.StageProduct
```

Other users in your workspace may have access to modify this item. Do not use this item unless you trust all users who may have access to the workspace.

(No column name)
11

00:00:05 Query executed successfully.

Load Data Warehouse
1 Hr 26 Min Remaining

Load data from a data lake by using the COPY statement

1. In your SQL script, enter the following code into the window.

```sql
SELECT COUNT(1)
FROM dbo.StageProduct
```

2. On the toolbar, use the ▷ **Run** button to run the SQL code and confirm that there are **0** rows currently in the **StageProduct** table.
3. Replace the code with the following COPY statement (changing **datalake*xxxxxx*** to the name of your data lake):

```sql
COPY INTO dbo.StageProduct
    (ProductID, ProductName, ProductCategory, Co
FROM 'https://datalakexxxxxx.blob.core.windows.r
WITH
(
    FILE_TYPE = 'CSV',
    MAXERRORS = 0,
    IDENTITY_INSERT = 'OFF',
    FIRSTROW = 2 --Skip header row
);


SELECT COUNT(1)
FROM dbo.StageProduct
```

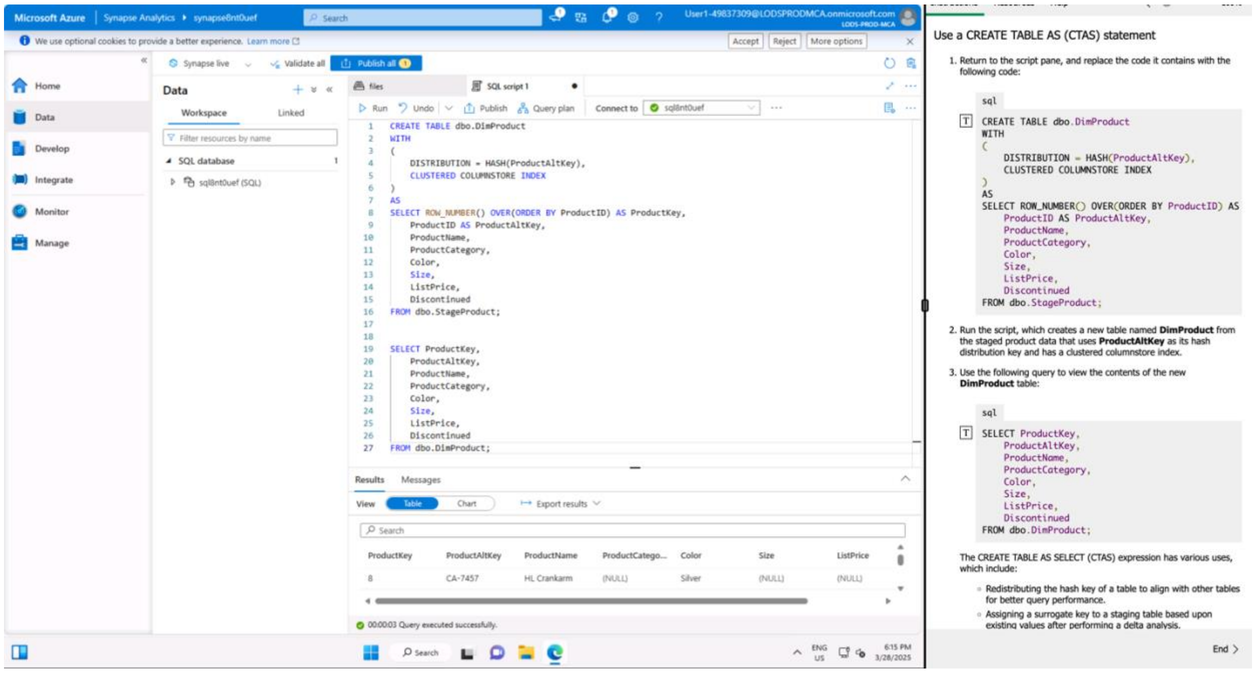
4. Run the script and review the results. 11 rows should have been loaded into the **StageProduct** table.

   Now let's use the same technique to load another table, this time logging any errors that might occur.

5. Replace the SQL code in the script pane with the following code, changing **datalake*xxxxxx*** to the name of your data lake in both the `FROM` and the `ERRORFILE` clauses:

```sql
COPY INTO dbo.StageCustomer
(GeographyKey, CustomerAlternateKey, Title, Firs
MaritalStatus, Suffix, Gender, EmailAddress, Yec
SpanishEducation, FrenchEducation, EnglishOccupc
NumberCarsOwned, AddressLine1, AddressLine2, Phc
FROM 'https://datalakexxxxxx.dfs.core.windows.ne
```

Using a CREATE TABLE AS (CTAS) statement:



```sql
CREATE TABLE dbo.DimProduct
WITH
(
    DISTRIBUTION = HASH(ProductAltKey),
    CLUSTERED COLUMNSTORE INDEX
)
AS
SELECT ROW_NUMBER() OVER(ORDER BY ProductID) AS ProductKey,
    ProductID AS ProductAltKey,
    ProductName,
    ProductCategory,
    Color,
    Size,
    ListPrice,
    Discontinued
FROM dbo.StageProduct;


SELECT ProductKey,
    ProductAltKey,
    ProductName,
    ProductCategory,
    Color,
    Size,
    ListPrice,
    Discontinued
FROM dbo.DimProduct;
```

ProductKey	ProductAltKey	ProductName	ProductCatego...	Color	Size	ListPrice
8	CA-7457	HL Crankarm	(NULL)	Silver	(NULL)	(NULL)

00:00:03 Query executed successfully.

Use a CREATE TABLE AS (CTAS) statement

1. Return to the script pane, and replace the code it contains with the following code:

```sql
CREATE TABLE dbo.DimProduct
WITH
(
    DISTRIBUTION = HASH(ProductAltKey),
    CLUSTERED COLUMNSTORE INDEX
)
AS
SELECT ROW_NUMBER() OVER(ORDER BY ProductID) AS
    ProductID AS ProductAltKey,
    ProductName,
    ProductCategory,
    Color,
    Size,
    ListPrice,
    Discontinued
FROM dbo.StageProduct;
```

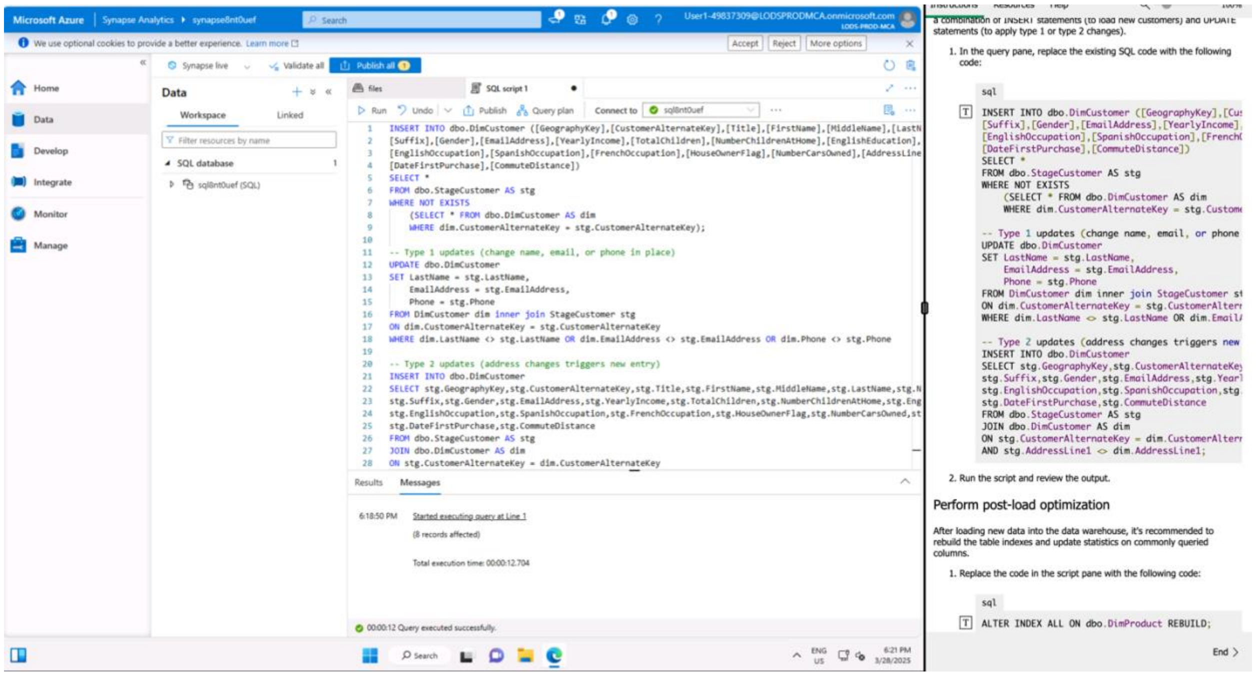
2. Run the script, which creates a new table named **DimProduct** from the staged product data that uses **ProductAltKey** as its hash distribution key and has a clustered columnstore index.
3. Use the following query to view the contents of the new **DimProduct** table:

```sql
SELECT ProductKey,
    ProductAltKey,
    ProductName,
    ProductCategory,
    Color,
    Size,
    ListPrice,
    Discontinued
FROM dbo.DimProduct;
```

The CREATE TABLE AS SELECT (CTAS) expression has various uses, which include:

- Redistributing the hash key of a table to align with other tables for better query performance.
- Assigning a surrogate key to a staging table based upon existing values after performing a delta analysis.

Combining INSERT and UPDATE statements to load a slowly changing dimension table:



```sql
INSERT INTO dbo.DimCustomer ([GeographyKey],[CustomerAlternateKey],[Title],[FirstName],[MiddleName],[LastN
[Suffix],[Gender],[EmailAddress],[YearlyIncome],[TotalChildren],[NumberChildrenAtHome],[EnglishEducation],
[EnglishOccupation],[SpanishOccupation],[FrenchOccupation],[HouseOwnerFlag],[NumberCarsOwned],[AddressLine
[DateFirstPurchase],[CommuteDistance])
SELECT *
FROM dbo.StageCustomer AS stg
WHERE NOT EXISTS
    (SELECT * FROM dbo.DimCustomer AS dim
    WHERE dim.CustomerAlternateKey = stg.CustomerAlternateKey);

-- Type 1 updates (change name, email, or phone in place)
UPDATE dbo.DimCustomer
SET LastName = stg.LastName,
    EmailAddress = stg.EmailAddress,
    Phone = stg.Phone
FROM DimCustomer dim inner join StageCustomer stg
ON dim.CustomerAlternateKey = stg.CustomerAlternateKey
WHERE dim.LastName <> stg.LastName OR dim.EmailAddress <> stg.EmailAddress OR dim.Phone <> stg.Phone

-- Type 2 updates (address changes triggers new entry)
INSERT INTO dbo.DimCustomer
SELECT stg.GeographyKey,stg.CustomerAlternateKey,stg.Title,stg.FirstName,stg.MiddleName,stg.LastName,stg.N
stg.Suffix,stg.Gender,stg.EmailAddress,stg.YearlyIncome,stg.TotalChildren,stg.NumberChildrenAtHome,stg.Eng
stg.EnglishOccupation,stg.SpanishOccupation,stg.FrenchOccupation,stg.HouseOwnerFlag,stg.NumberCarsOwned,st
stg.DateFirstPurchase,stg.CommuteDistance
FROM dbo.StageCustomer AS stg
JOIN dbo.DimCustomer AS dim
ON stg.CustomerAlternateKey = dim.CustomerAlternateKey
```

```
6:18:50 PM    Started executing query at Line 1
              (8 records affected)

              Total execution time: 00:00:12.704

00:00:12 Query executed successfully.
```

a combination of INSERT statements (to load new customers) and UPDATE statements (to apply type 1 or type 2 changes).

1. In the query pane, replace the existing SQL code with the following code:

```sql
INSERT INTO dbo.DimCustomer ([GeographyKey],[Cu
[Suffix],[Gender],[EmailAddress],[YearlyIncome],
[EnglishOccupation],[SpanishOccupation],[French(
[DateFirstPurchase],[CommuteDistance])
SELECT *
FROM dbo.StageCustomer AS stg
WHERE NOT EXISTS
    (SELECT * FROM dbo.DimCustomer AS dim
    WHERE dim.CustomerAlternateKey = stg.Custom

-- Type 1 updates (change name, email, or phone
UPDATE dbo.DimCustomer
SET LastName = stg.LastName,
    EmailAddress = stg.EmailAddress,
    Phone = stg.Phone
FROM DimCustomer dim inner join StageCustomer st
ON dim.CustomerAlternateKey = stg.CustomerAlter
WHERE dim.LastName <> stg.LastName OR dim.Email

-- Type 2 updates (address changes triggers new
INSERT INTO dbo.DimCustomer
SELECT stg.GeographyKey,stg.CustomerAlternateKey
stg.Suffix,stg.Gender,stg.EmailAddress,stg.Year
stg.EnglishOccupation,stg.SpanishOccupation,stg
stg.DateFirstPurchase,stg.CommuteDistance
FROM dbo.StageCustomer AS stg
JOIN dbo.DimCustomer AS dim
ON stg.CustomerAlternateKey = dim.CustomerAlter
AND stg.AddressLine1 <> dim.AddressLine1;
```

2. Run the script and review the output.

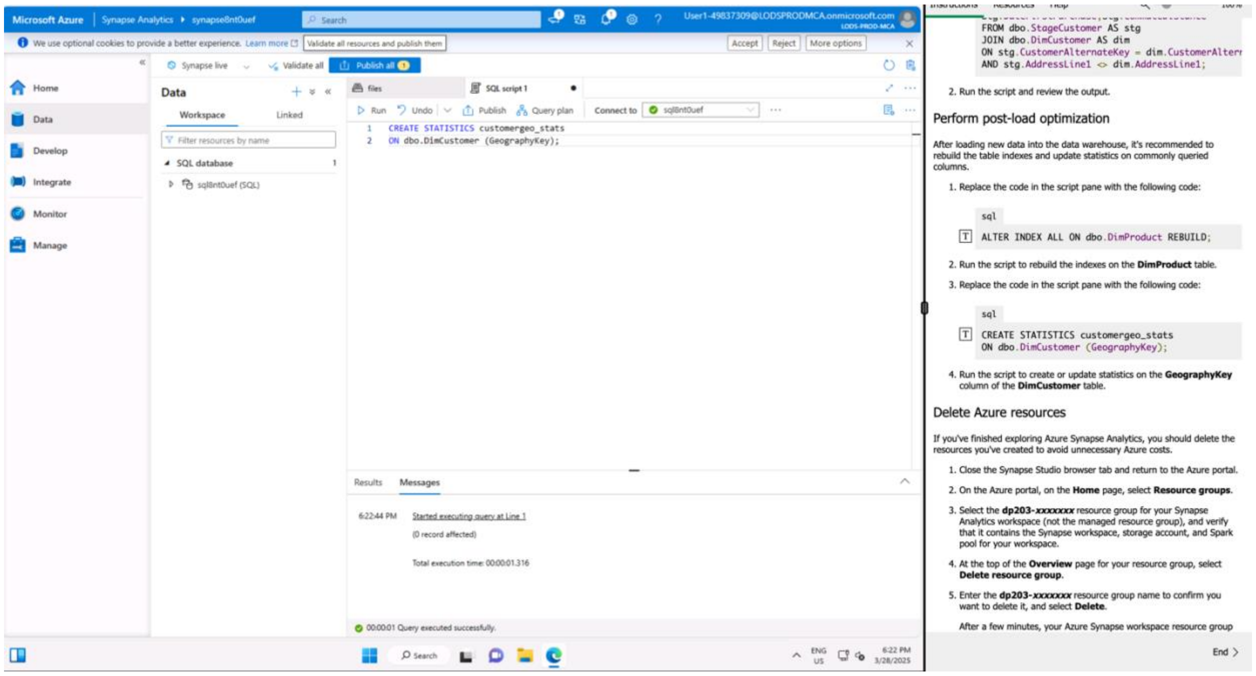
Perform post-load optimization

After loading new data into the data warehouse, it's recommended to rebuild the table indexes and update statistics on commonly queried columns.

1. Replace the code in the script pane with the following code:

```sql
ALTER INDEX ALL ON dbo.DimProduct REBUILD;
```

Performing post-load optimization:



```sql
CREATE STATISTICS customergeo_stats
ON dbo.DimCustomer (GeographyKey);
```

```
6:22:44 PM    Started executing query at Line 1
              (0 record affected)

              Total execution time: 00:00:01.316

00:00:01 Query executed successfully.
```

```sql
FROM dbo.StageCustomer AS stg
JOIN dbo.DimCustomer AS dim
ON stg.CustomerAlternateKey = dim.CustomerAlter
AND stg.AddressLine1 <> dim.AddressLine1;
```

2. Run the script and review the output.

Perform post-load optimization

After loading new data into the data warehouse, it's recommended to rebuild the table indexes and update statistics on commonly queried columns.

1. Replace the code in the script pane with the following code:

```sql
ALTER INDEX ALL ON dbo.DimProduct REBUILD;
```

2. Run the script to rebuild the indexes on the **DimProduct** table.
3. Replace the code in the script pane with the following code:

```sql
CREATE STATISTICS customergeo_stats
ON dbo.DimCustomer (GeographyKey);
```

4. Run the script to create or update statistics on the **GeographyKey** column of the **DimCustomer** table.

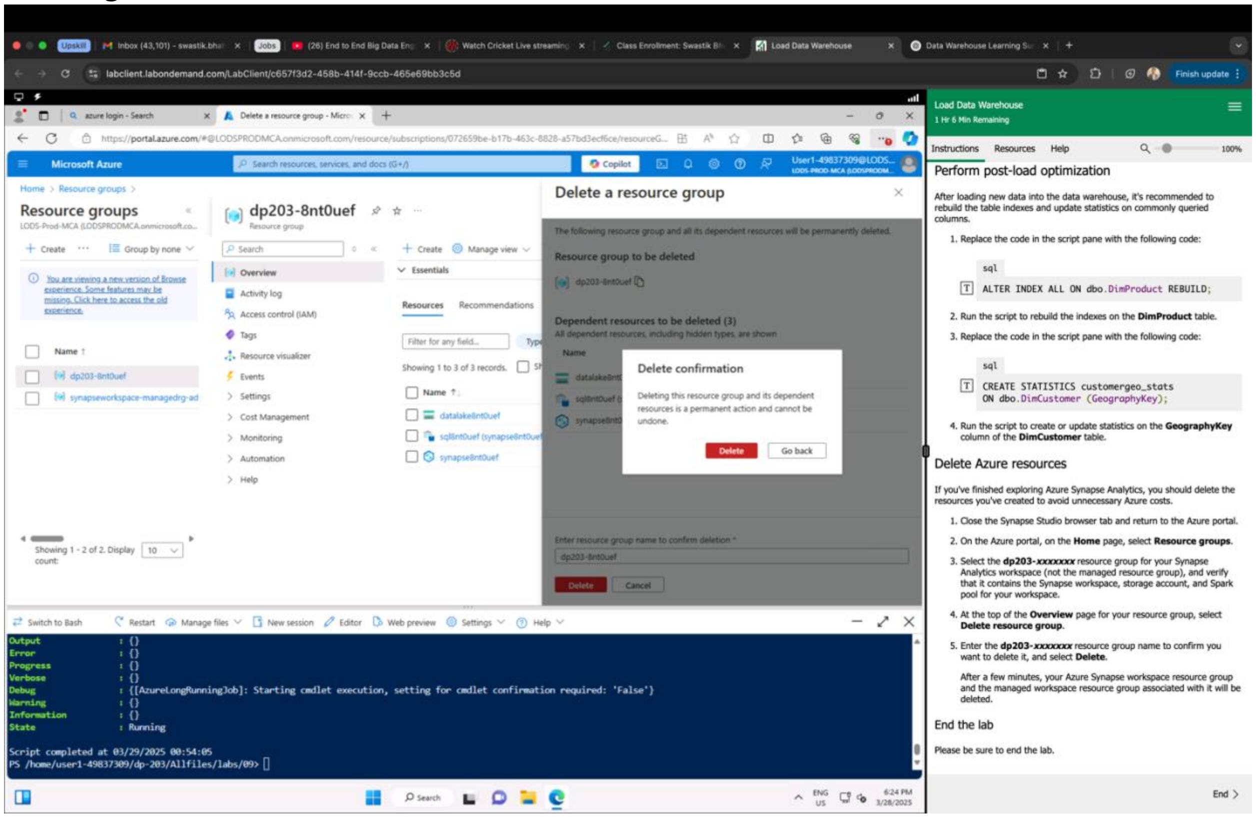
Delete Azure resources

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 Spark pool for your workspace.
4. At the top of the **Overview** page for your resource group, select **Delete resource group**.
5. 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

Deleting Azure resources:



Delete a resource group

The following resource group and all its dependent resources will be permanently deleted.

Resource group to be deleted: dp203-8nt0uef

Dependent resources to be deleted (3)

Delete confirmation

Deleting this resource group and its dependent resources is a permanent action and cannot be undone.

Enter resource group name to confirm deletion *: dp203-8nt0uef

```
Output              : {}
Error               : {}
Progress            : {}
Verbose             : {}
Debug               : {[AzureLongRunningJob]: Starting cmdlet execution, setting for cmdlet confirmation required: 'False'}
Warning             : {}
Information         : {}
State               : Running

Script completed at 03/29/2025 00:54:05
PS /home/user1-49837309/dp-203/Allfiles/labs/09> 
```

Load Data Warehouse
1 Hr 6 Min Remaining

Perform post-load optimization

After loading new data into the data warehouse, it's recommended to rebuild the table indexes and update statistics on commonly queried columns.

1. Replace the code in the script pane with the following code:

```sql
ALTER INDEX ALL ON dbo.DimProduct REBUILD;
```

2. Run the script to rebuild the indexes on the **DimProduct** table.
3. Replace the code in the script pane with the following code:

```sql
CREATE STATISTICS customergeo_stats
ON dbo.DimCustomer (GeographyKey);
```

4. Run the script to create or update statistics on the **GeographyKey** column of the **DimCustomer** table.

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 Spark pool for your workspace.
4. At the top of the **Overview** page for your resource group, select **Delete resource group**.
5. 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.

End the lab

Please be sure to end the lab.

Conclusion:

In this lab, I learned how to resume SQL pools, load data from a data lake using the COPY statement, and utilize the CREATE TABLE AS (CTAS) statement for efficient table creation. Additionally, I practiced combining INSERT and UPDATE statements to handle slowly changing dimension tables and performed post-load optimization. This process enhanced my understanding of data loading techniques and best practices for maintaining data integrity and optimizing performance within a data warehouse environment.