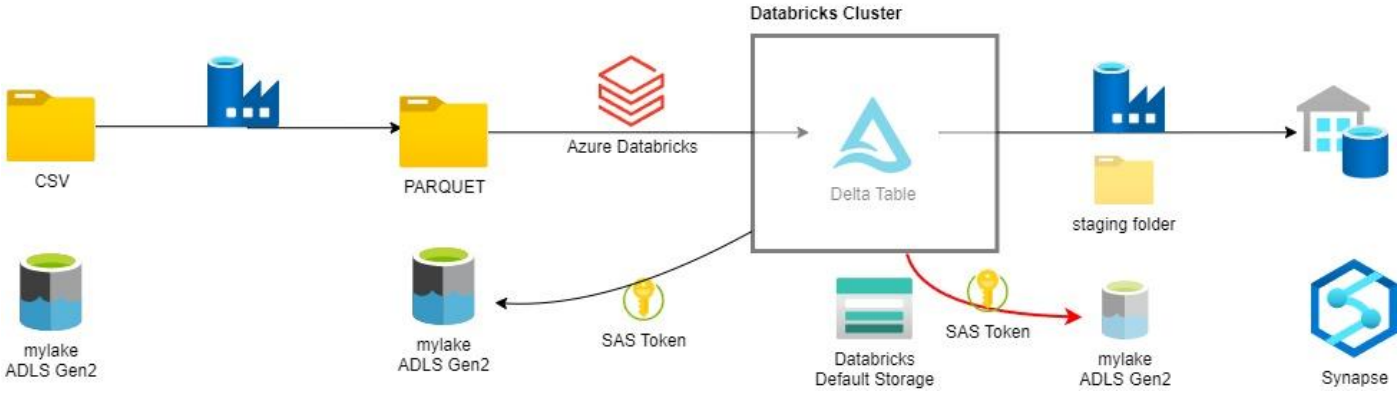
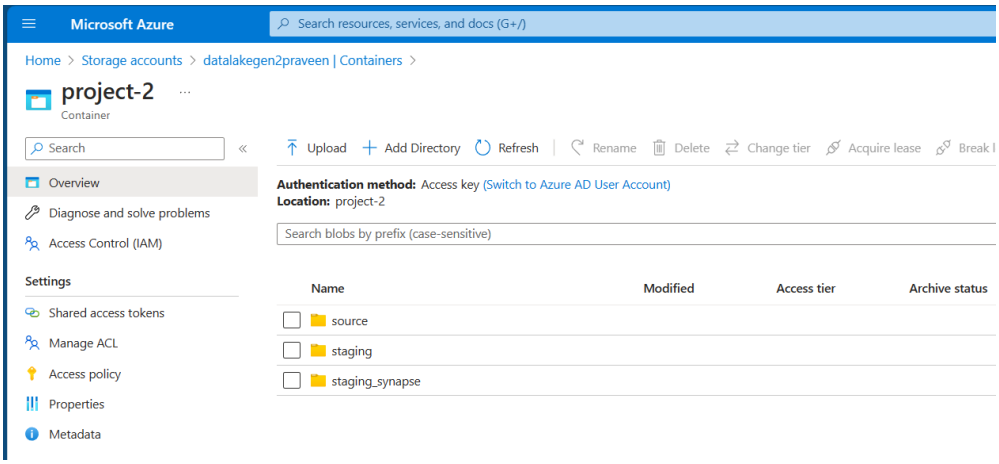
Task – Convert CSV files stored to Parquet, load into Delta Tables, then load data from Delta table into SQL database.

Architecture diagram:

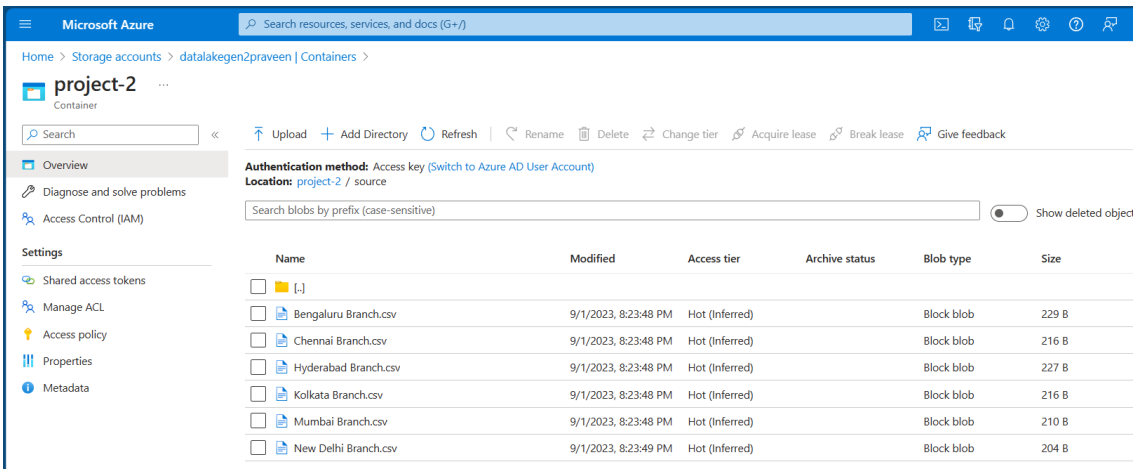


Part 1 – Convert CSV files to Parquet files.

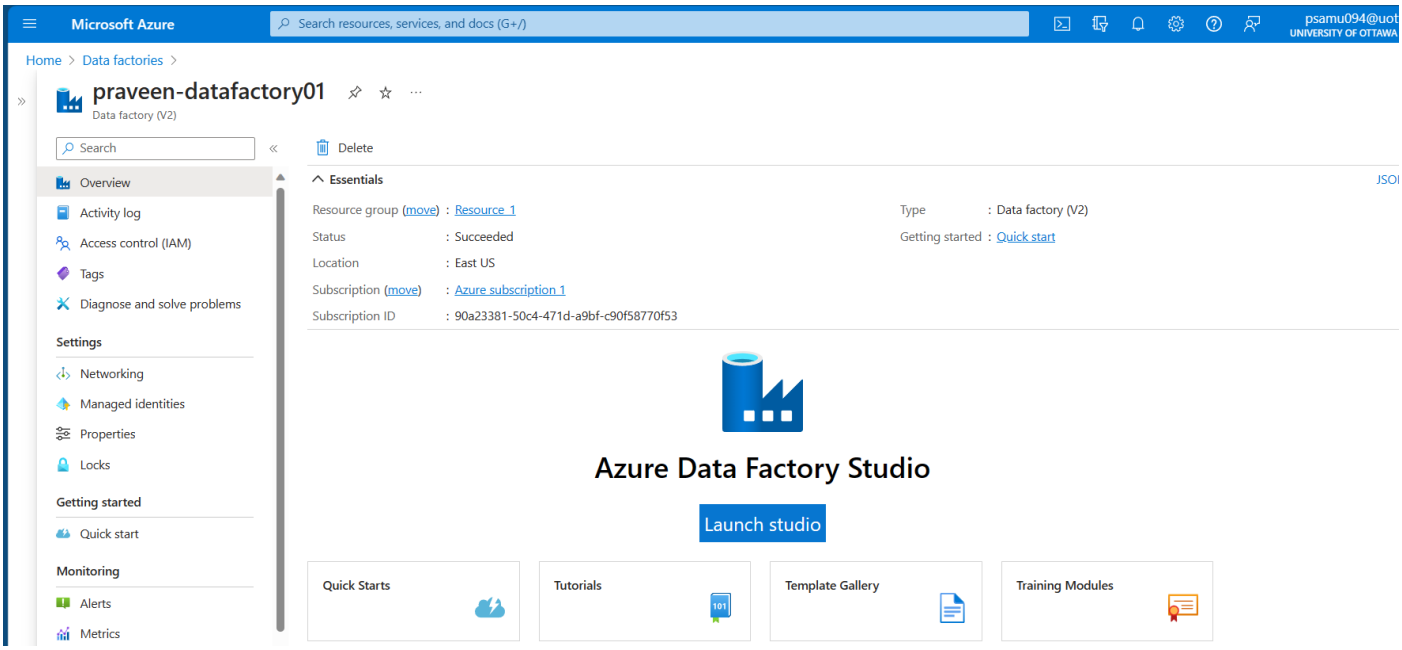
1. In the Azure Portal home page, search for 'Storage Account' and open the resource.
2. Click on Create.
3. In "Basics" page, under Project details, enter Subscription, resource group. Under Instance details, create a new name – 'datalakegen2praveen' for account, choose region and other relevant details.
4. Click 'Review' to review the account details and submit.
5. In the menu, choose 'Containers'. Click on + icon to create a new container. Give a name and click 'Create'.
6. Open the container, click '+ Add Directory' to create a folder. Create 3 such folders named 'Source', 'Staging', and 'staging_synapse'.



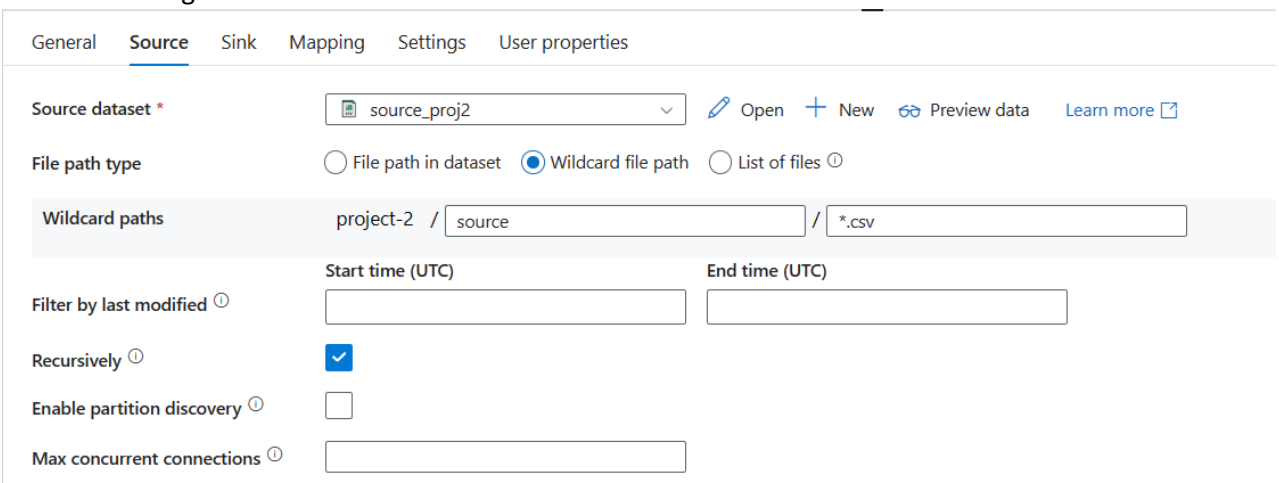
7. Open the 'source' folder, click on 'Upload' and upload the CSV files.



8. Search for ‘Azure Data Factory’ in the portal and open the Data Factory resource. Create a new resource, choose the subscription, resource group, and provide the instance details – created as ‘praveen-datafactory01’. Click on ‘Review and Create’, review the details and click on ‘Create’. Once the resource is created, click on ‘Launch Studio’

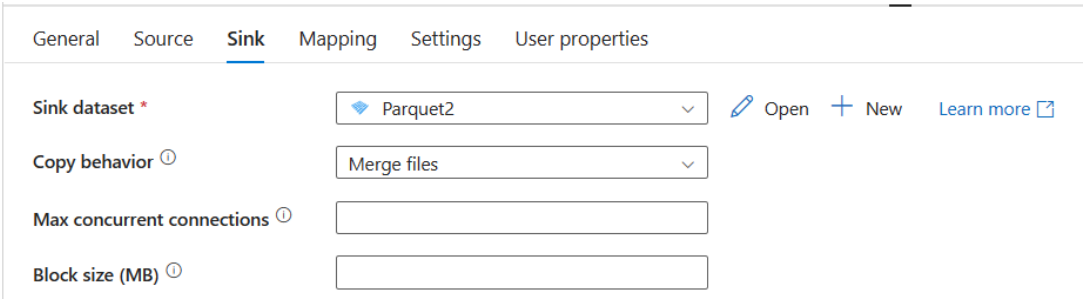


9. In the Data Factory Studio, open the ‘Manage’  page, click on ‘Linked Services’.
10. Click on ‘+’ icon to create a new linked service. Choose ‘Azure Data Lake Storage Gen 2’ as Data Store. Name it as ‘AzureDataLakeStorage2’. Choose the Azure Subscription and the storage account created in step 3 (‘datalakegen2praveen’). Click on ‘Test Connection’ to verify the linked service and click ‘Create’.
11. In the Data Factory Studio, open the ‘Author’  pipeline page. Click on ‘+’ icon -> Pipeline -> Pipeline to create a new pipeline.
12. Under Activities -> Move and Transform, drag the ‘Copy Data’ activity and drop onto the workspace.
13. Under General settings, name the activity as ‘csv_to_parquet’ and leave the remaining settings to default.
14. Under Source section, create new source dataset. ‘+ New’ icon -> Azure Data Lake Gen 2 -> Delimited Text as Format -> Click Continue. In the properties page, choose the linked service ‘AzureDataLakeStorage2’, choose the folder path of ‘source’ folder created in Data Lake. Remove the check ‘First row as header’. Click on ‘Okay’ to create the dataset.
15. Have the settings as indicated in screenshot below.

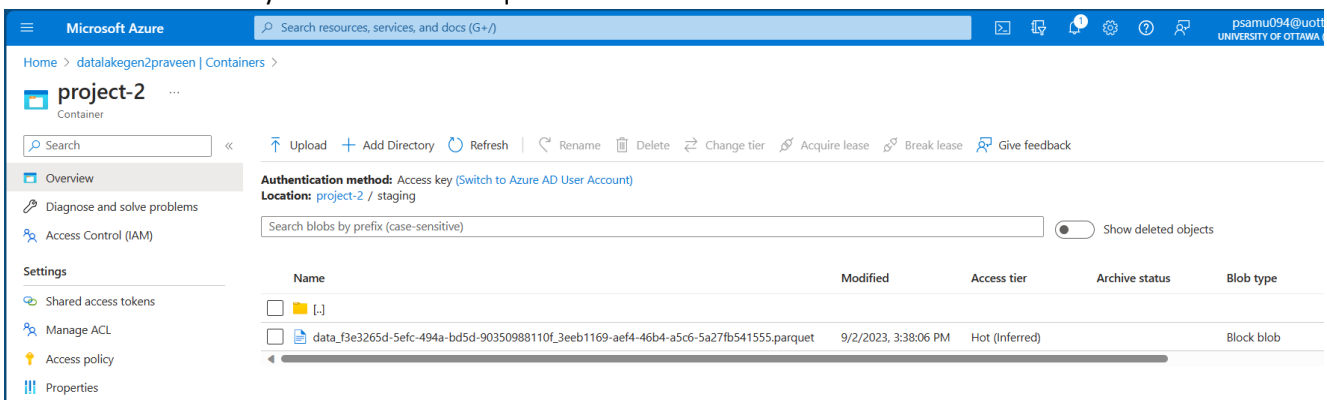


16. Under ‘Sink’ setting, click ‘+ New’ to create a new Sink dataset. Choose Azure Data Lake Gen 2 -> Parquet. Under linked service, choose the same as earlier ‘AzureDataLakeStorage2’ and choose the staging folder in ADLS as file path.

17. Have the sink settings as shown in below screenshot.

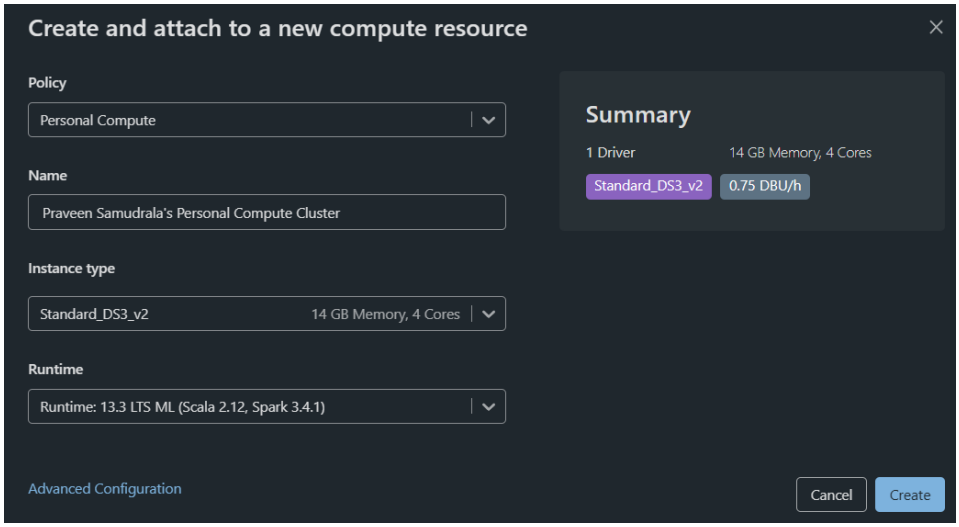


18. Publish the pipeline and Trigger now to run it. Monitor it's progress and once successful, open the staging folder in ADLS to verify the converted Parquet file.



Step 2 – Load the parquet file into a Delta Table.

1. In Azure homepage portal, search for 'Azure Databricks' and open the resource. Create a new Azure Databricks resource by clicking on '+' icon. Choose the subscription and resource group. Enter the instance details with name as 'praveenadbspace01'. Review and Submit the request.
2. Once the resource us deployed, open the resource and click on Launch Workspace.
3. Create a new notebook 'project_2' and create a new compute cluster Praveen Personal Compute Cluster' with details as shown below '.



4. In a new tab, open ADLS storage account created earlier i.e 'datalakegen2praveen', from the menu, open 'Access Keys' under 'Security + Monitoring' section. Copy the Key.
5. Now in Azure Databricks, open 'Compute' from menu on left side, open the newly created cluster, click 'Edit' to edit the configuration. Open 'Advanced Options', in the spark config box, paste the following code.

   fs.azure.account.key.datalakegen2praveen.dfs.core.windows.net storage_access_key

   replace the 'storage_access_key' with the ADLS account access key copied in earlier step.
   With this, we are providing the Spark cluster the access to ADLS storage account for reading the files, and also writing to use as a staging layer in later stage.

6. Go to the notebook, change the name to ‘project2-notebook’, and enter the code to read the merged parquet file.

```
# Reading parquet frile from ADLS and saving as Delta Table
for x in dbutils.fs.ls("abfss://project-2@datalakegen2praveen.dfs.core.windows.net/staging"):
    df = spark.read.parquet(x.path)
    display(df)
    df.write.format('delta').mode('overwrite').saveAsTable('sales')
```

(11) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [Location: string, Brand_Name: string ... 2 more fields]

Table +

	Location	Brand_Name	Sales_Count	Revenue
1	Hyderabad	Peter England	300	170000
2	Delhi	Peter England	220	110000
3	Mumbai	Peter England	300	170000
4	Kolkata	Peter England	100	80000
5	Chennai	Peter England	300	220000
6	Bengaluru	Peter England	320	200000
7	Delhi	Fab India	50	25000

```
# Read the Delta table
df_delta = spark.read.load('dbfs:/user/hive/warehouse/sales')
display(df_delta)
```

(2) Spark Jobs

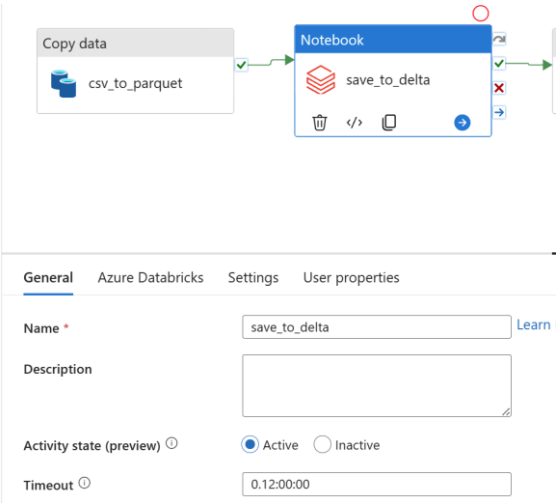
df_delta: pyspark.sql.dataframe.DataFrame = [Location: string, Brand_Name: string ... 2 more fields]

Table +

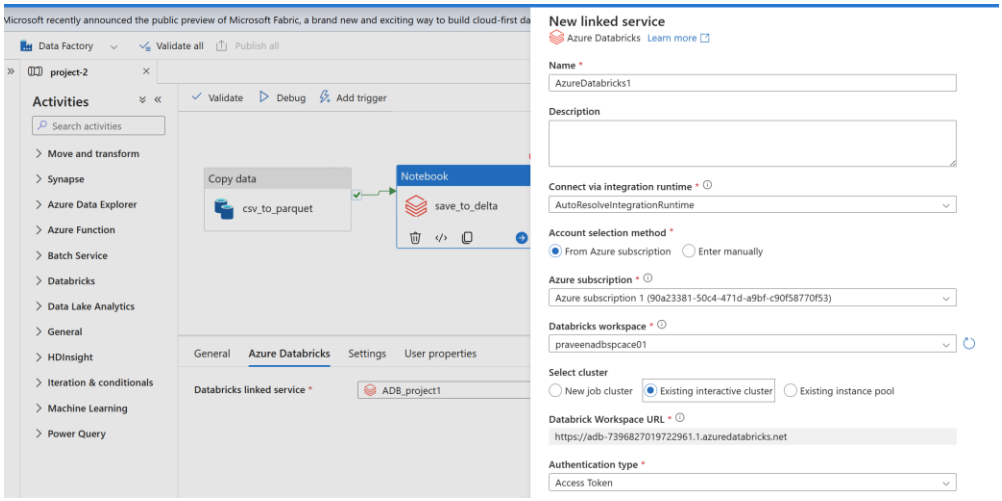
	Location	Brand_Name	Sales_Count	Revenue
1	Hyderabad	Peter England	300	170000
2	Delhi	Peter England	220	110000
3	Mumbai	Peter England	300	170000
4	Kolkata	Peter England	100	80000
5	Chennai	Peter England	300	220000
6	Bengaluru	Peter England	320	200000
7	Delhi	Fab India	50	25000

36 rows | 0.79 seconds runtime Refreshed

7. Run the complete notebook to check if code is running free of errors.
8. Now in Azure Databricks resource, click on the account name in top-right corner in the resource -> User Settings -> Developer -> Manage against Access tokens. Generate a new token and copy the access token.
9. Open Azure Data Factory resource and open the pipeline that is being worked on.
10. Add a new activity after Copy ‘csv_to_parquet’ step, choose ‘Notebook’ under Databricks section. Name the step as ‘save_to_delta’ in general section.



11. In ‘Azure Databricks’ section, click ‘+New’ to create a new Databricks Linked Service. Enter the relevant details as shown in screenshot.

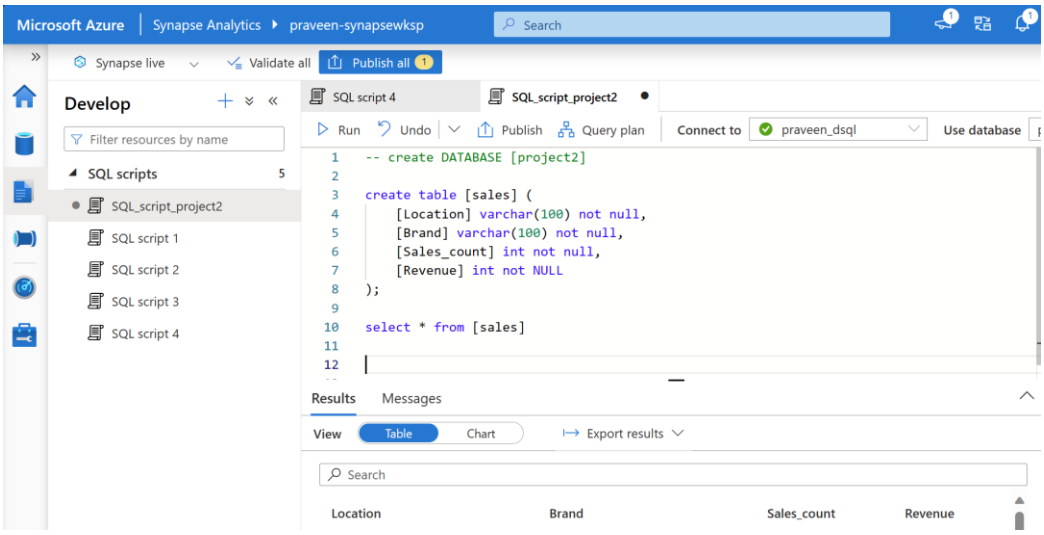


Paste the access token copied from notebook above and choose the cluster 'Praveen Cluster'. Test the connection and click 'Create'.

12. In the 'Notebook' section, browse through and choose the notebook created earlier 'project2-notebook'.
13. Publish the changes and Trigger the pipeline to it's working.

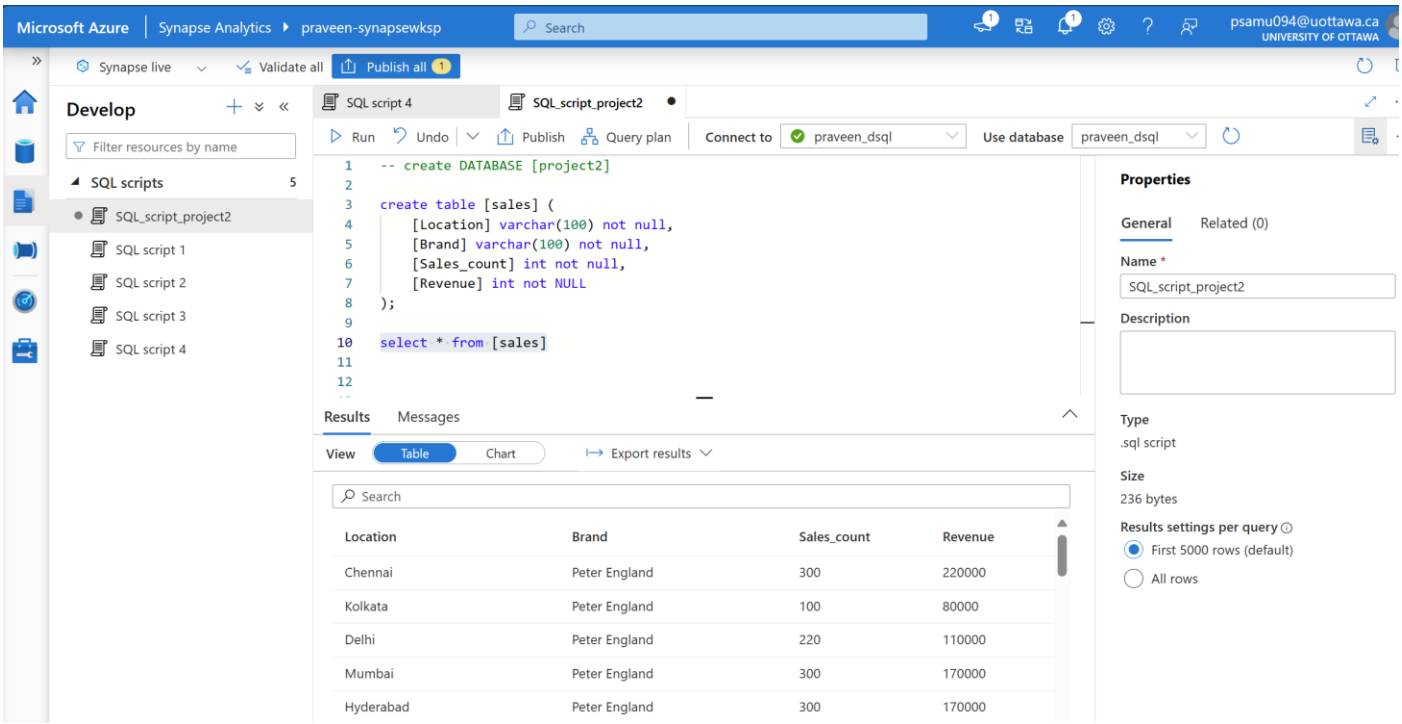
Step 3 – Load Delta table into Synapse SQL DB.

1. In the search bar of Azure portal, search for 'Azure Synapse' and open the resource. Click on 'Create Synapse Workspace'. Choose the subscription, resource group, enter workspace name as 'praveen_synapsewksp' and choose a ADLS storage account. In the 'Security' tab, enter the SQL server admin username and password. Review the details and create the resource.
2. Open the resource and create a new Dedicated SQL Pool with name as 'praveen_dsql' and wait until the resource is created.
3. Open Synapse Studio, click on the  icon to open the SQL script development page.
4. Choose the created sql pool 'praveen_dsql' under the dropdown against cluster and enter the below code to create a table with schema aligning with the data.



5. Run the script to see an empty table and publish the changes.
6. Now, in the Data Factory resource, create a new Linked Service with Data Store as Azure Databricks Delta Lake. Enter the relevant details – subscription, databricks workspace, access keys and cluster ID. Test the connection and create the linked service.
7. Create another new Linked Service with Data Store as 'Azure Synapse Analytics'. Enter the subscription, server name, database name, username and password for the database, test the connection and create the linked service.

8. In the edit pipeline page, add a new Copy activity after Databricks Notebook step and name it as ‘Copy_Delta_SynapseSQL’.
9. In the ‘Source’ section, create a new source dataset with data store as Delta Lake and choose the LinkedService created earlier for Delta tables, database as ‘default’ and table as ‘sales’.
10. In the ‘Sink’ section, create a new sink dataset with data store as Azure Synapse Analytics and choose the linked service create above for Synapse, choose the table name as ‘dbo.sales’ and create the dataset. Choose the copy method as ‘Copy Command’ and leave other values to defaults.
11. In ‘Mapping’ section, click on ‘Import Schema’ to fetch schema from both Delta and Synapse tables and map the columns and datatypes.
12. In ‘Settings’ section, enable the check box against ‘Enable Staging’. For staging account linked service, choose the linked service created for ADLS earlier and storage path as the path for ‘staging_synapse’ folder in the container, and leave other settings to default.
13. Publish the changes and trigger the pipeline to run now.
14. Once the pipeline is successfully run, open the Synapse studio and query ‘sales’ table to see the result.



<u>Possibility of issues:</u>

1. Run all the cells in databricks notebook to identify and issues such as table and data already existing. Enable overwriting of data into delta table.
2. Authentication errors such as invalid Key error(Invalid configuration value detected for fs.azure.account.key). Configure the Spark cluster with access to ADLS.
3. Column not found in SQL DW table. Mapping of column names of Delta table should be done with column names of Synapse SQL Table. Not to worry if enabled mapping in Step 3.11.