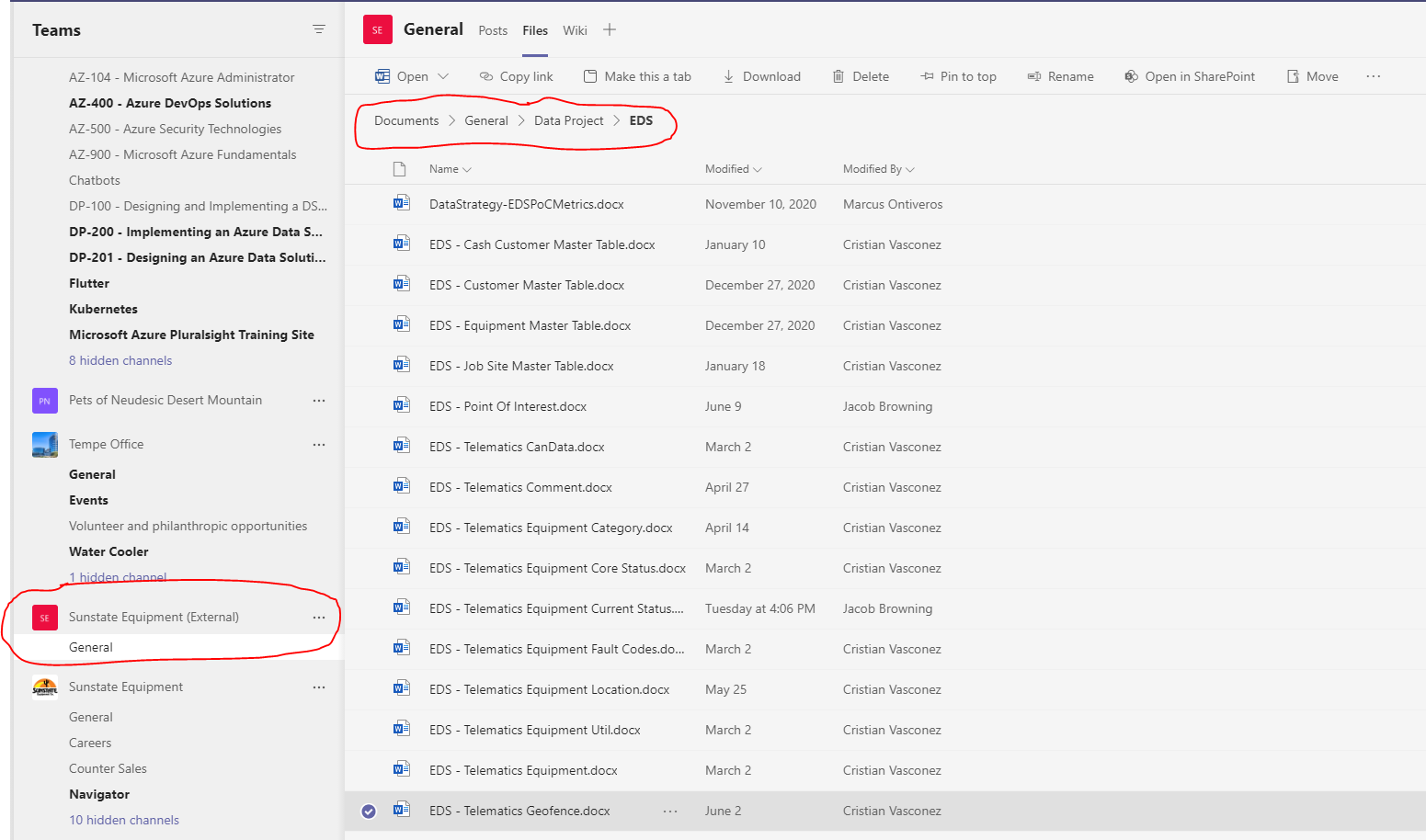
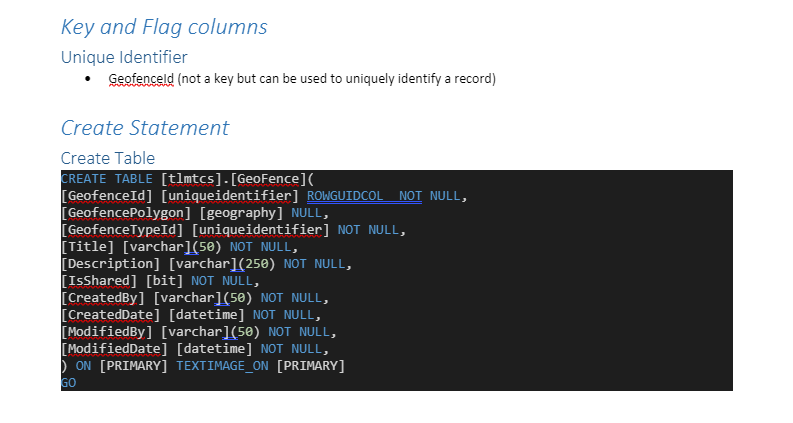
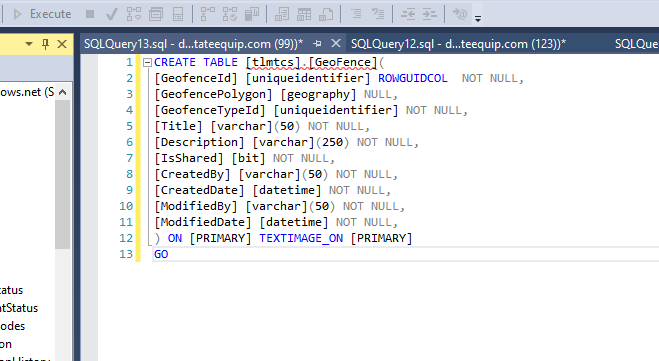
**CDC End-to-End process**

1. Create tables from Create Statement
   1. Open the data mapping document from the **Sunstate Equipment (External)** team

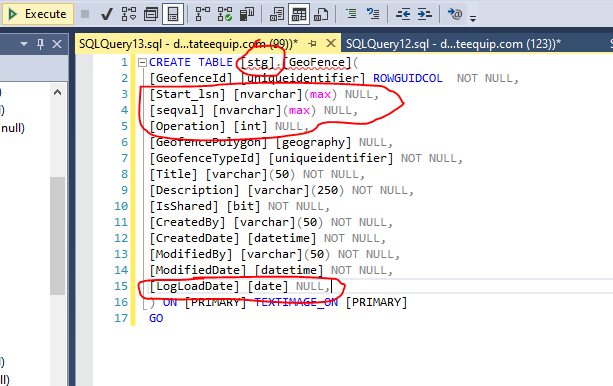


* 1. Copy/paste the Create into new query

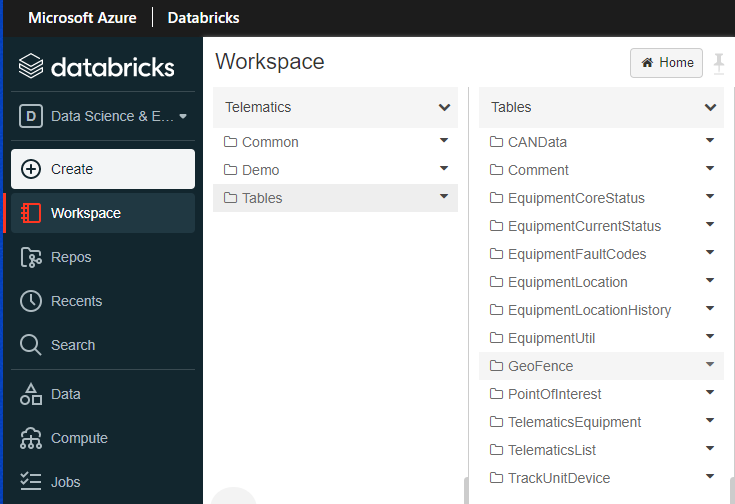




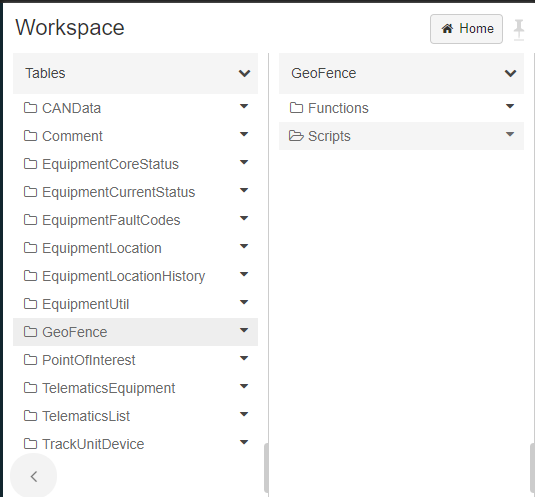
1. Change tlmtcs to stg
   1. Add four extra columns:
      1. Start\_lsn
      2. seqval
      3. Operation
      4. Comp
   2. Remove primary key constraints (if any)

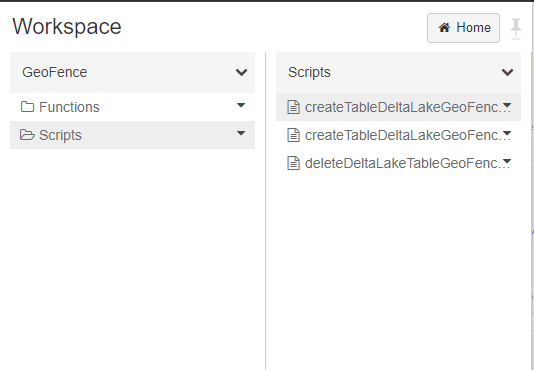


1. Create Delta Tables for new object
   1. Create new table folder in CDC/Telematics/Tables directory in Databricks



* 1. Create a Scripts subfolder in your new table folder



* 1. Clone CreateTableDeltaLake(TableName) script notebook from any existing table folder into your new table folder. Replace (TableName) with the name of your object
  2. Replace all table names in **entire notebook** with new table name



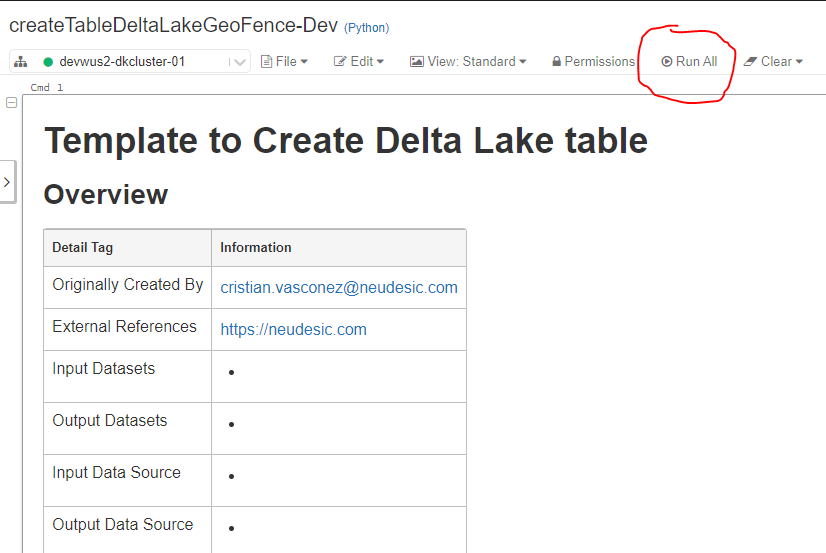
* 1. Replace column names with names from new table
     1. Add *update\_mask* and *Status* to **audit** and **sql\_logs** tables



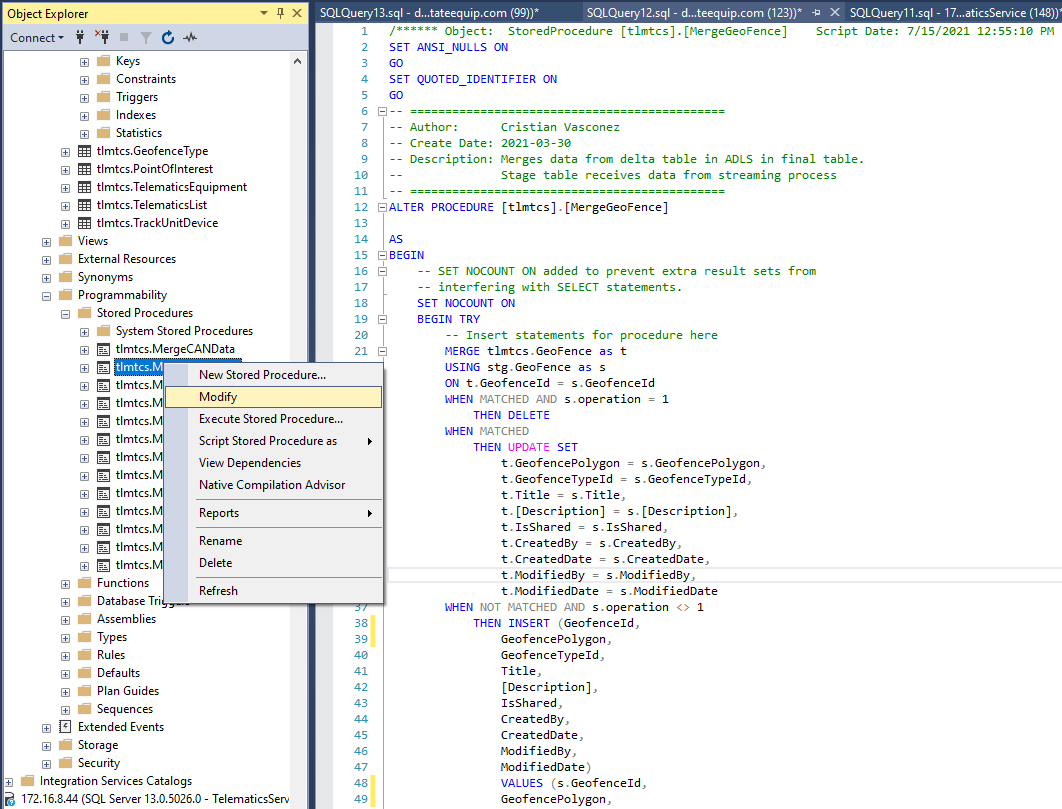
* + 1. For raw table, make sure to indicate if column is NULL or not based on SQL schema



* 1. Once all tables are updated with correct table names and columns, click Run All at the top of the notebook

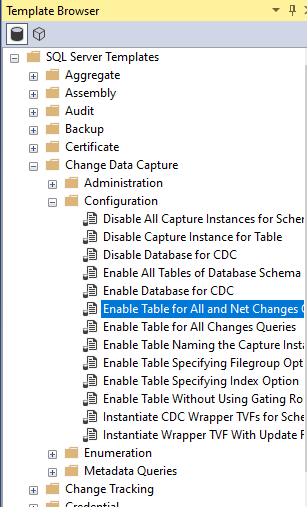


1. Create stored procedure in SMSS
   1. In the **devwus2-mdp-sql01.database.windows.net** database, Modify any existing stored procedure (Databases-> dev-wus2mdp-sbd01->Programmability->Stored Procedures

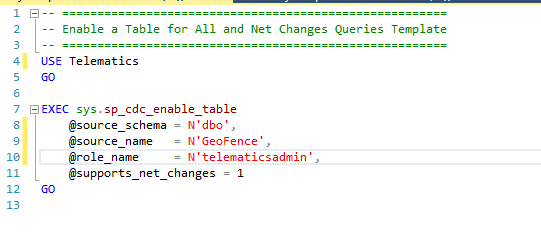


* 1. Edit stored procedure to use table name and columns from the object you are preparing.
     1. Make sure **ALTER** is changed to **CREATE**
     2. Make sure primary key columns are used in ON statement
  2. Execute query to create stored procedure

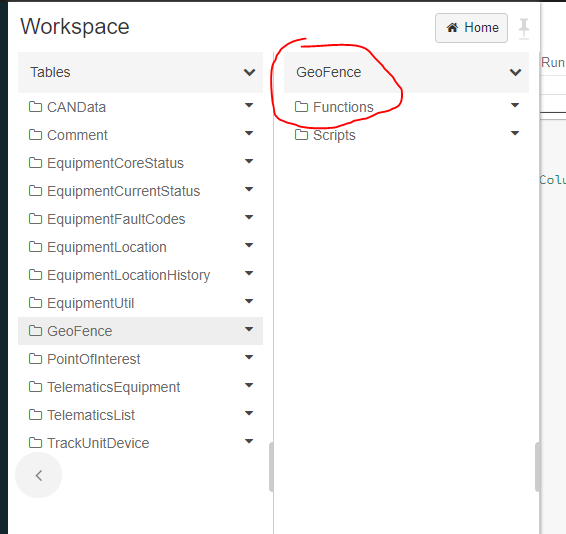
1. Enable cdc in sql server
   1. In SSMS, open the Template Explorer from View->Template Explorer
   2. Select the **Enable a Table for All and Net Changes Queries Template** from the Template Explorer



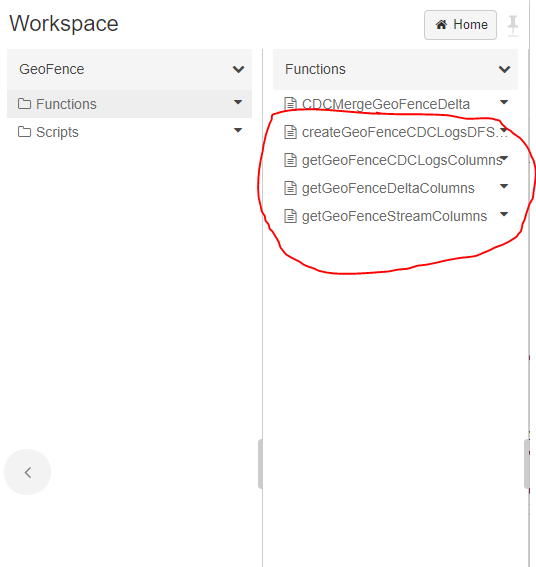
* 1. Fill out the template with the relevant parameters
     1. USE = Telematics
     2. @source\_schema = N’dbo’
     3. @source\_name = N’[Table Name]’
     4. @role\_name = N’telematicsadmin’
     5. @supports\_net\_changes = 1



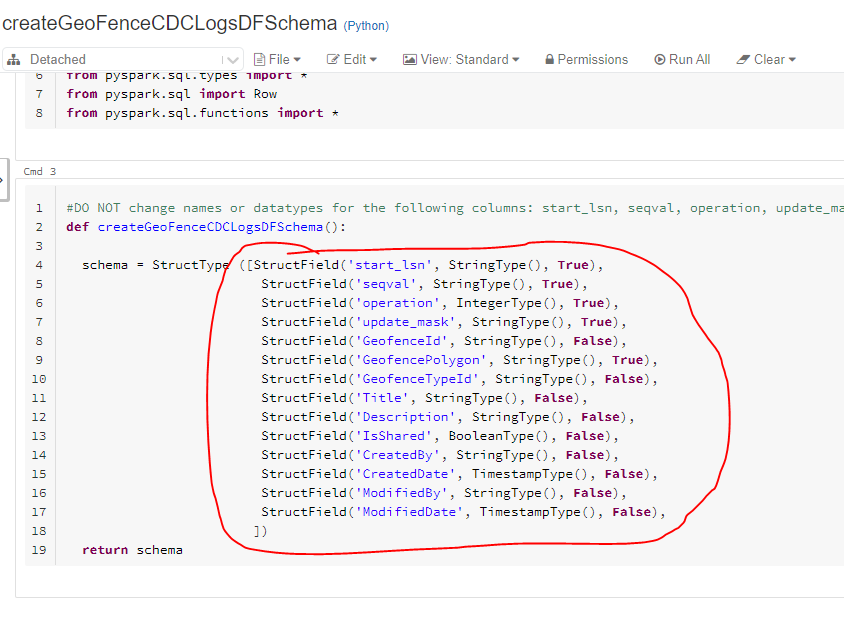
1. Create Delta Table functions for object in Databricks
   1. Create a Functions folder under your table’s directory



* 1. Clone the four columns notebooks to create schemas for delta tables from any existing table
     1. Create[TableName]CDCLogsDFSchema
     2. get[TableName]CDCLogsColumns
     3. get[TableName]DeltaColumns
     4. get[TableName]StreamColumns

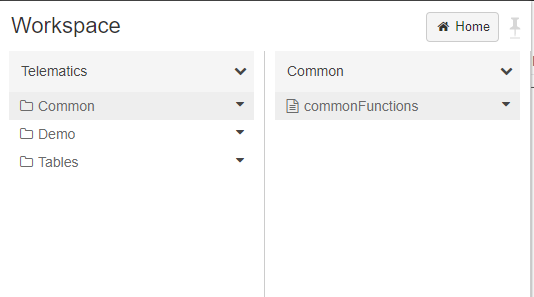


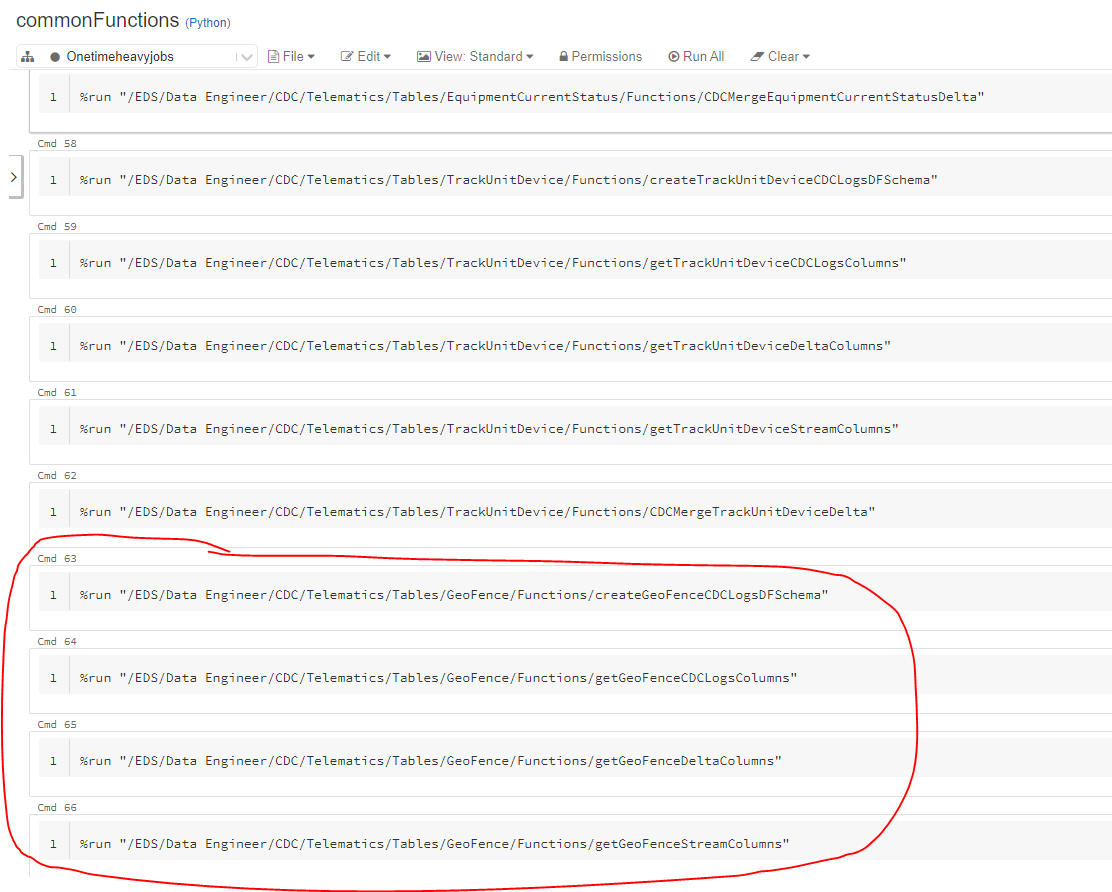
* 1. Replace contents of each notebook with the relevant table names and column names



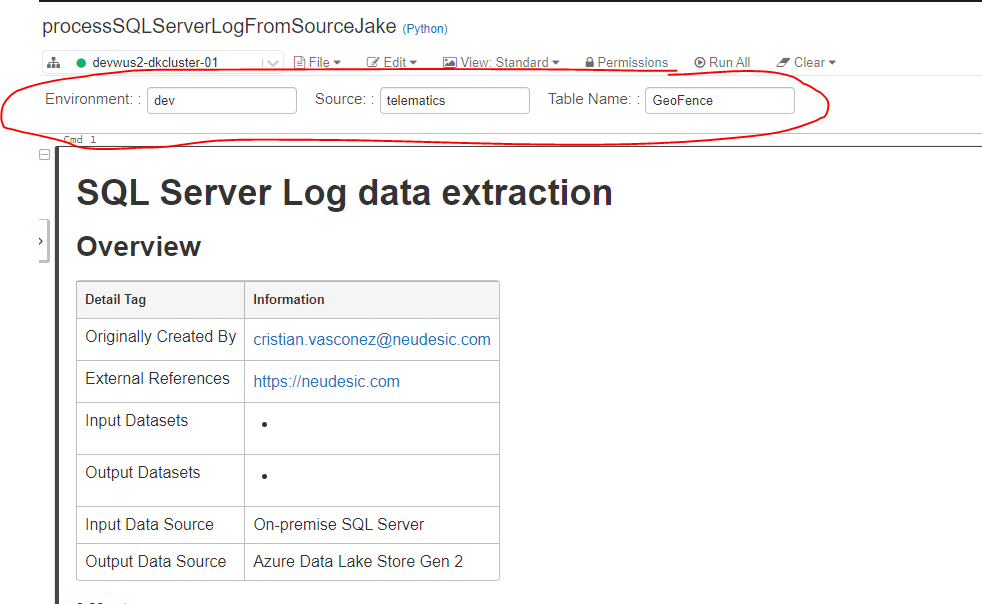
*Repeat concept with other three Functions notebooks*

1. Add new functions to Common Functions folder as new cells

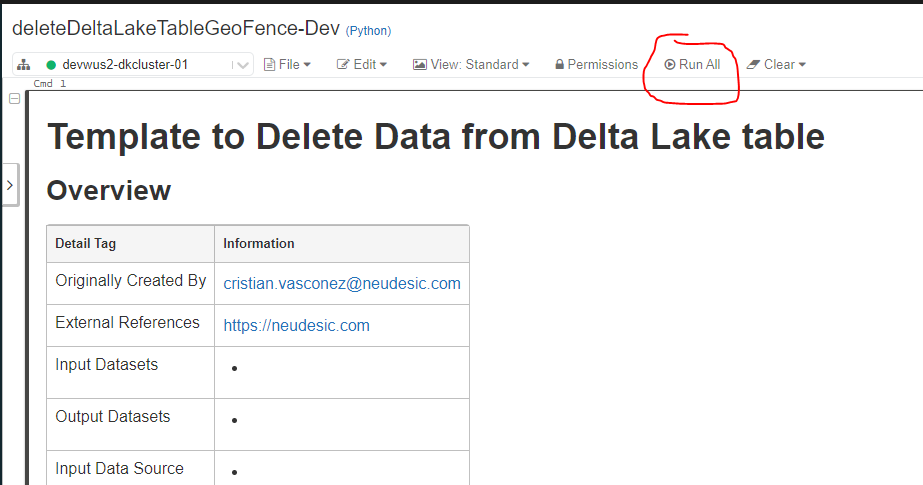




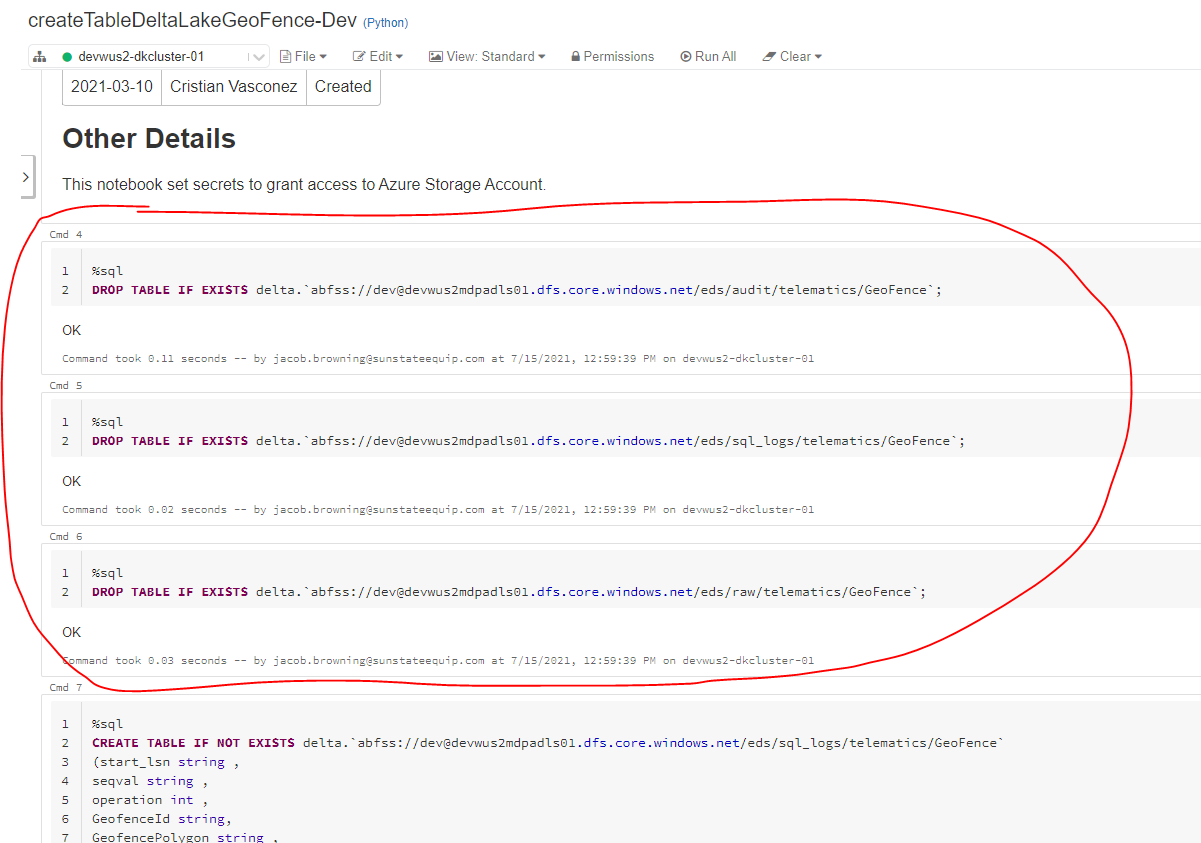
1. Run processSQLServerLogFromSource notebook
   1. Clone notebook from Troobleshoot-Testing location in CDC directory to your personal Databricks directory (accessed from root directory)
   2. Set the appropriate parameters at the top of the notebook



* 1. If data log is too large (EquipmentCoreStatus), use Onetimeheavyjobs cluster instead of devwus2-dkcluster-01 custer.
  2. Click Run All at top of notebook
  3. **If notebook Fails at any point:**
     1. Run All in deleteDeltaLakeTable[TableName] in Scripts folder for your table. If notebook doesn’t exist, clone from existing table and replace table names with your table.



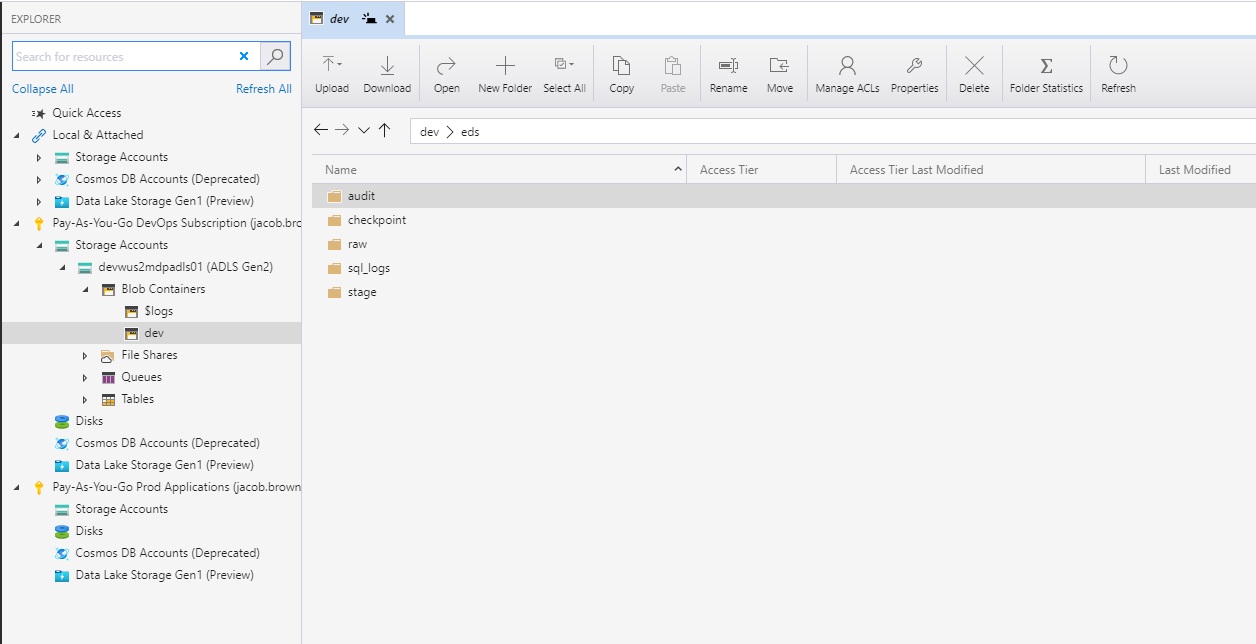
* + 1. Drop the table manually in createTableDeltaLake[TableName] using cells 4-6

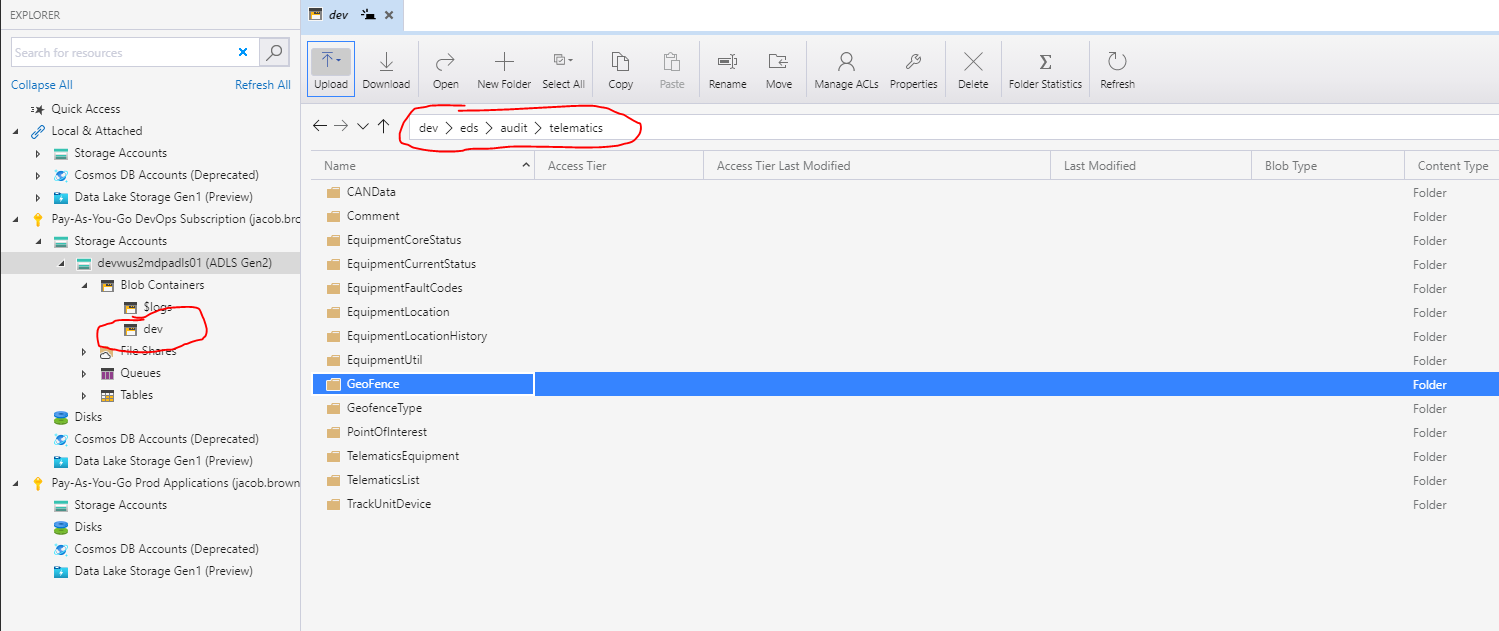


* + 1. Delete the delta logs manually in **dewwus2mdpadls01** Azure Storage account *(requires Azure Storage Explorer application – download here:* <https://azure.microsoft.com/en-us/features/storage-explorer/>)

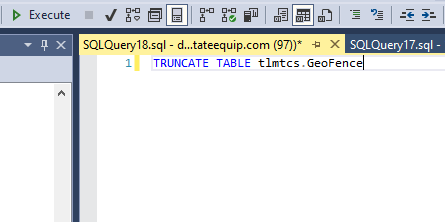
Delete table folder from following locations in blob storage:

* audit
* checkpoint (if Stream process already run)
* sql\_logs
* raw

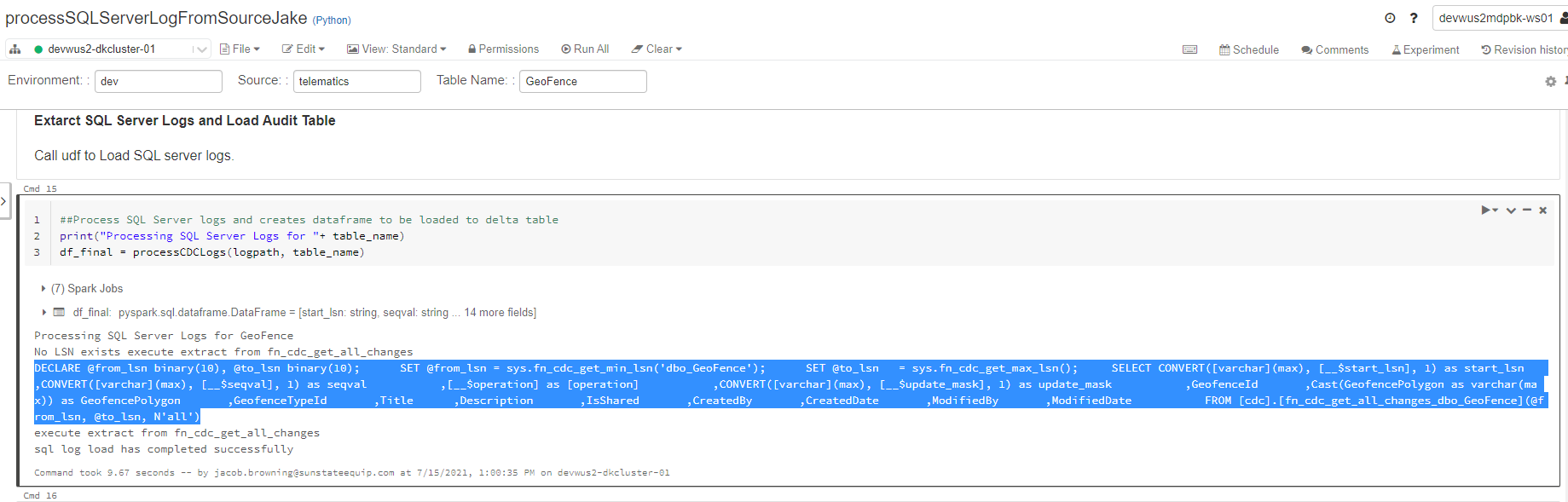




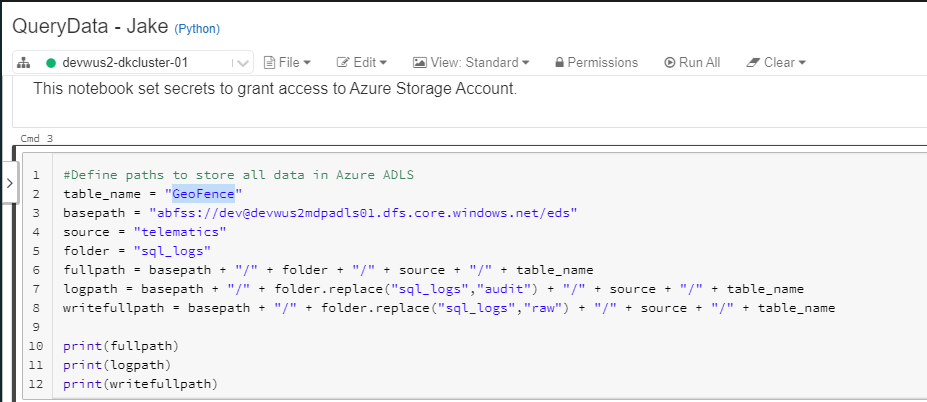
* + 1. If destination table [TableName].tlmtcs in SQL has data, use TRUNCATE TABLE query to clear out old data



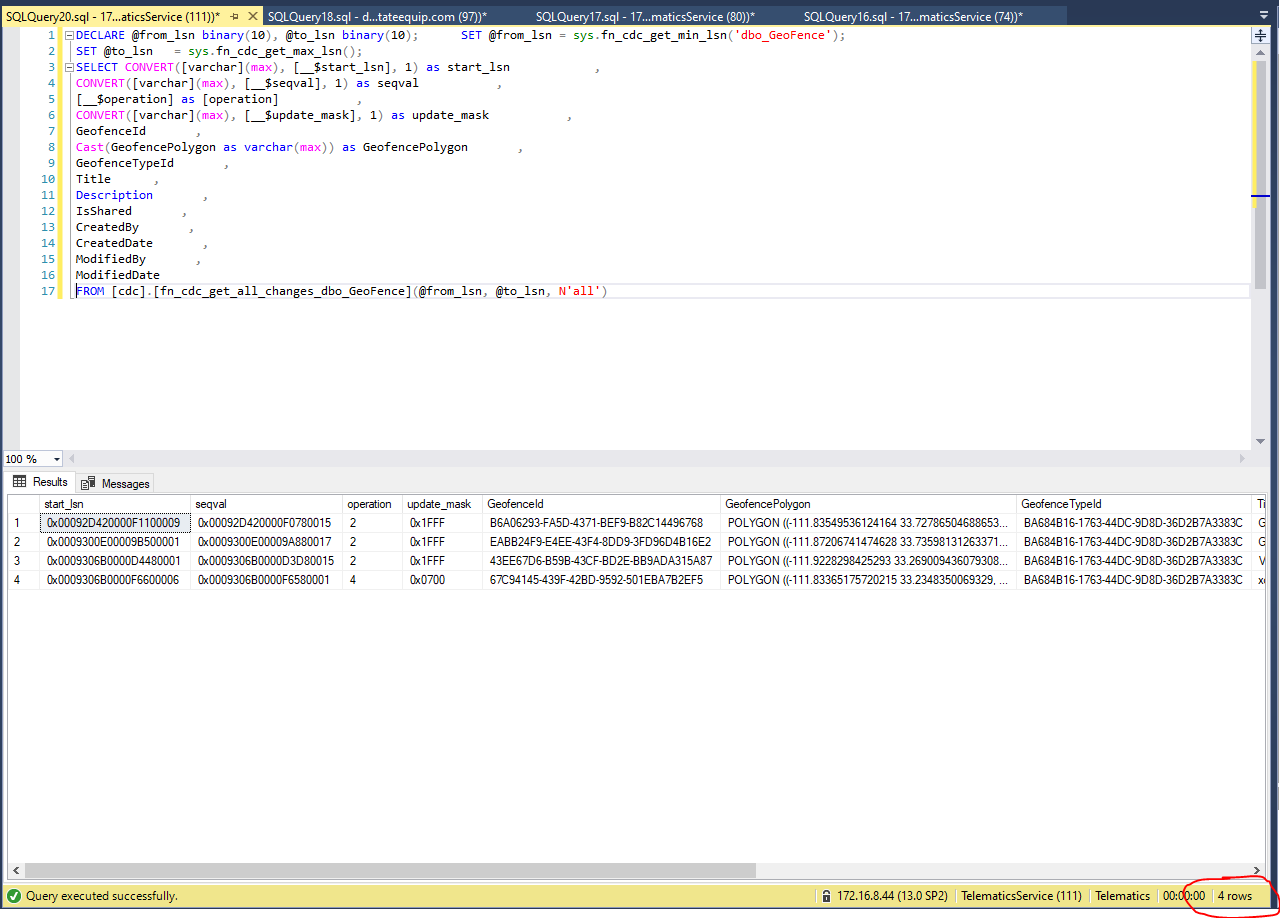
1. Validate CDC process data with actual data in SMSS
   1. Copy query from processSQLServerLogs output in Cell 15

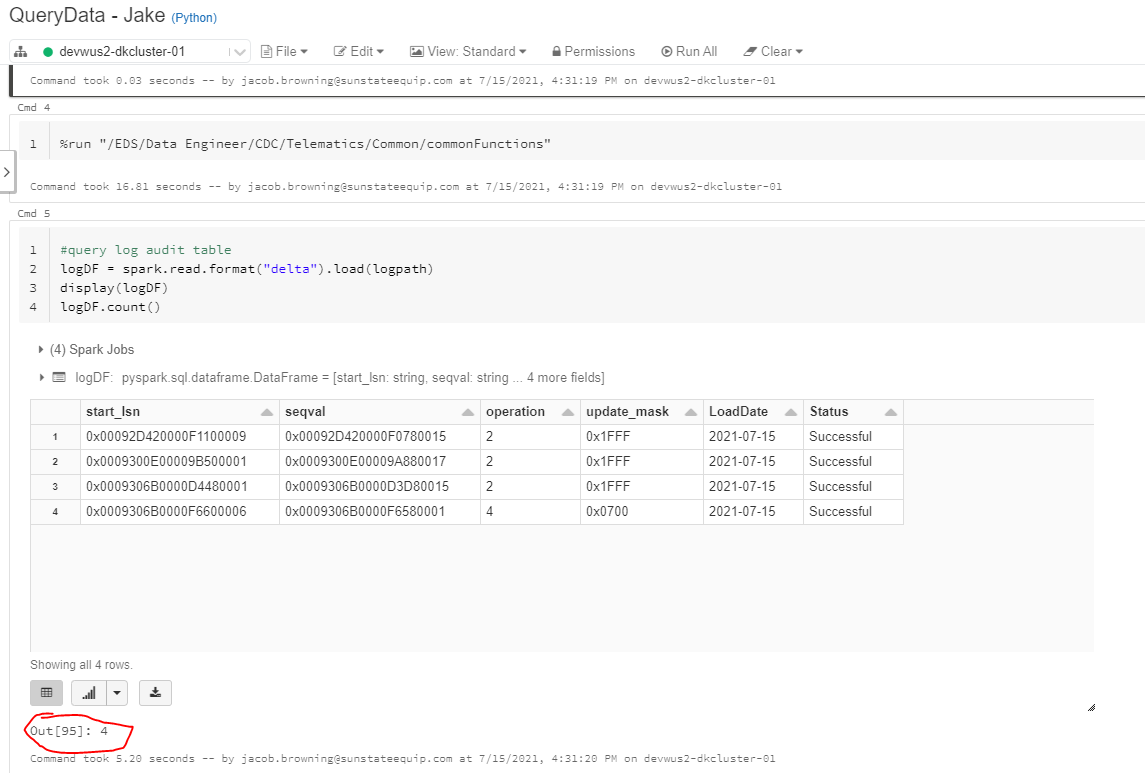


* 1. Paste into a query on the Telematics database in ssevmteldevdb.sunstate.com server
  2. Clone the QueryData notebook to your personal environment (if not done already)
  3. Change the **table\_name in line 2 of Cell 3** to your table name

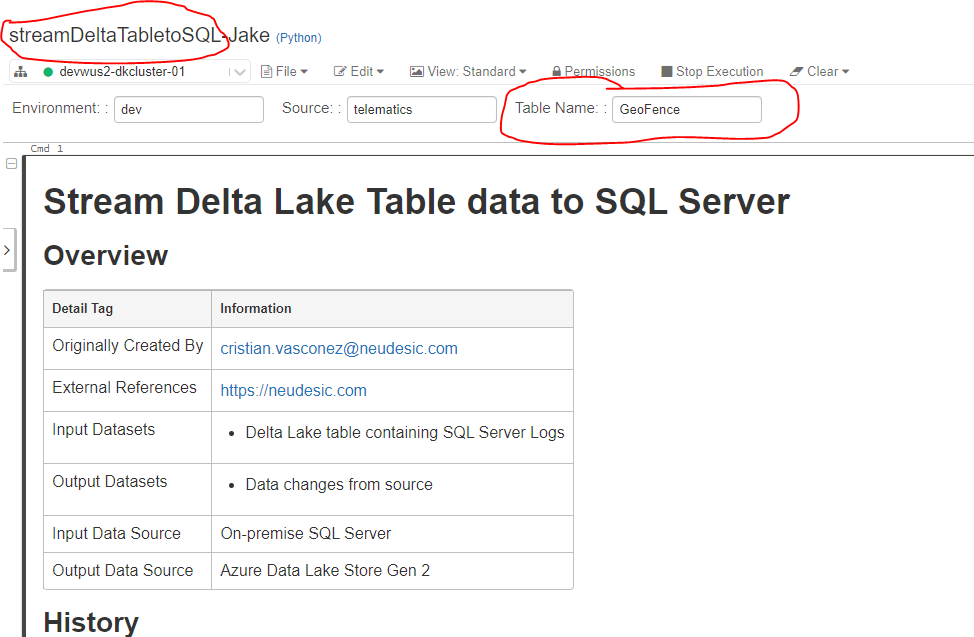


* 1. Run QueryData notebook to make sure that the number of rows in the 5th cell match the number of rows from the pasted SQL query

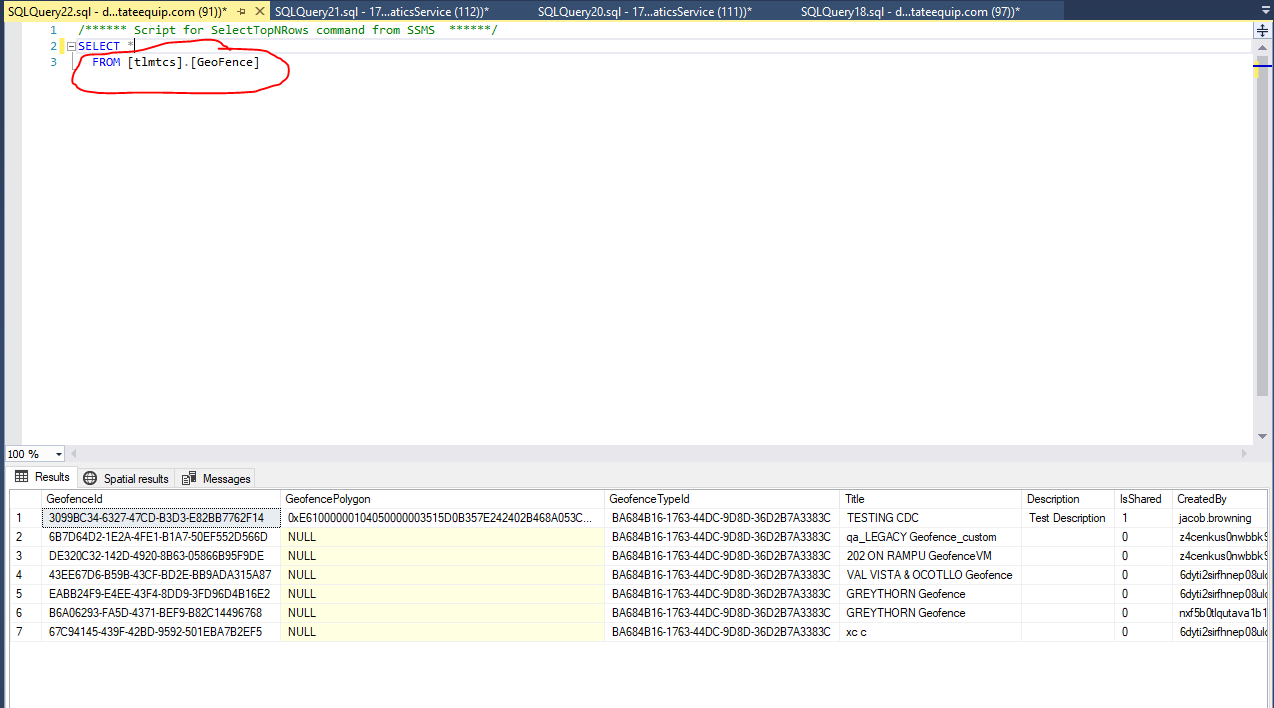




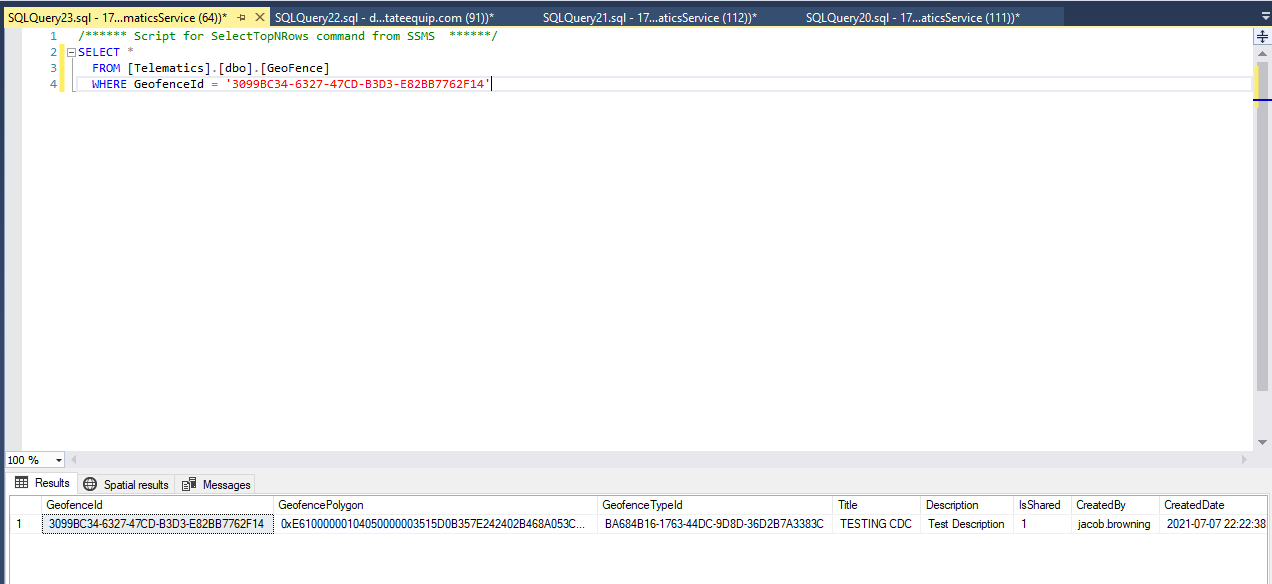
1. Run streamDeltaTabletoSQL to enable streaming process
   1. Clone streamDeltaTabletoSQL notebook from CDC/Notebooks (if not done already)
   2. Set table parameter to current table
   3. Click Run All at top



1. Verify data has been copied over to telematics version of table your created in SQL
   1. Run SELECT \* on the tlmtcs.[TableName] destination table in devwus2-mdp-sql01.database.windows.net



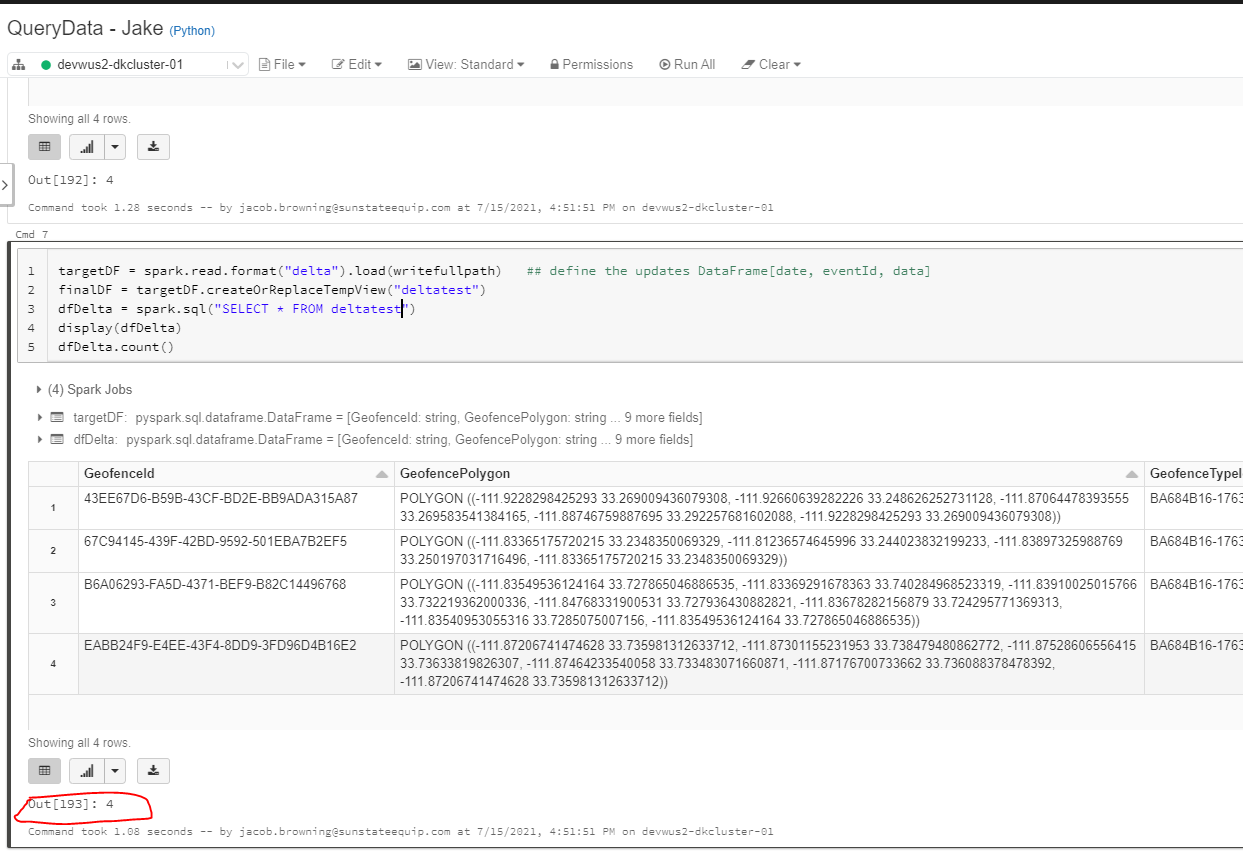
* 1. Choose a row of data, select on the source Telematics.dbo.[TableName] table in Telematics on the unique identifier(s)



* 1. Compare each column of data between the destination and source row. Check for any discrepencies including
     1. Decimal precision
     2. Capitalization of strings
     3. DateTime values
     4. NULL columns

**\*Note**: Some columns from the source Telematics.dbo.[TableName] table might be missing in the destination tlmtcs.[TableName] table. Confirm if the column has a *REMOVED* transformation in the data mapping document referred to in Step 1.

* 1. Confirm that the number of rows in Cell 7 of QueryData notebook matches the number of rows in tlmtcs.[TableName].
     1. Check the latest *ModifiedDate* value, make sure it’s the same as latest



Pushing to DevOps:

1. Check in notebook to Main
2. Create initial commit comment and save (if new notebook)

Troubleshooting:

processCDCLogs