**Azure Storage Explorer and Delta Live Table Assignment**

**Name: Tejaswini Gokanakonda - DE142**

**Date: 03-12-2024**

**Create a storage explorer step by process to extract data on adls**

**Step 1: Install Azure Storage Explorer**

1. **Download Azure Storage Explorer**:
   * Visit the [official Azure Storage Explorer website](https://azure.microsoft.com/en-us/products/storage/storage-explorer/).
   * Download the installer for your operating system (Windows, macOS, or Linux).
2. **Install the Tool**:
   * Follow the installation steps and launch Azure Storage Explorer after installation.

A person sitting at a table with a computer

Description automatically generated

**Step 2: Create a Storage Account in Azure Cloud**

1. **Sign in to Azure Portal**:
   * Navigate to [Azure Portal](https://portal.azure.com/) and log in with your credentials.
2. **Create a New Storage Account**:
   * Go to **Storage Accounts** in the left-hand menu and click **+ Create**.
   * Fill in the required details:
     + Subscription: Select your subscription.
     + Resource Group: Select or create a new resource group.
     + Storage Account Name: Provide a unique name.
     + Performance: Choose between Standard or Premium.
     + Redundancy: Select your preferred redundancy option (e.g., LRS, GRS).
   * Click **Review + Create** and then **Create**.

**Step 3: Set Up Azure Data Lake Storage (ADLS)**

1. **Enable Hierarchical Namespace**:
   * Navigate to your newly created storage account in the Azure Portal.
   * Go to **Configuration** under **Settings** and enable **Hierarchical Namespace** for ADLS Gen2.
   * Save the changes.
2. **Create a Container**:
   * Under the **Data Storage** section, click on **Containers**.
   * Click **+ Container**, provide a name, and set access level (Private, Blob, or Container).

A screenshot of a computer

Description automatically generated

**Step 4: Generate a Shared Access Token (SAS)**

1. **Generate SAS Token**:
   * Navigate to your container in the Azure Portal.
   * Select **Shared access signature** under the **Settings** section.
   * Specify permissions:
     + Check **Read**, **Write**, **Delete**, **List**, and other necessary permissions.
   * Set the **Expiry Date** to a future date as needed.
   * Click **Generate SAS and connection string**.

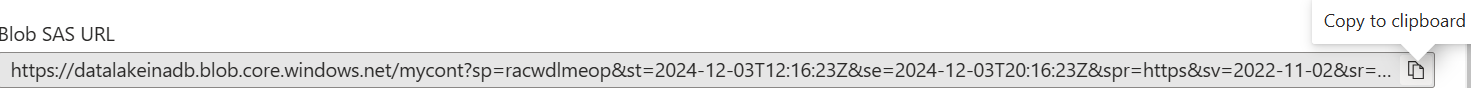
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. **Save the SAS URL**:
   * Copy the **Blob SAS URL** and paste it into a notepad for later use.



**Step 5: Sign In to Azure Storage Explorer**

1. Open Azure Storage Explorer and click **Sign In**.
2. Enter your Azure credentials to connect Storage Explorer to your Azure account.
3. Once signed in, your subscriptions and storage accounts will be displayed.

A screenshot of a computer login

Description automatically generated



**Step 6: Connect to ADLS Using SAS URL**

1. **Use the Connect Option**:
   * In the left-hand dashboard, click **Connect**.
   * Select **Use a Shared Access Signature (SAS) URI**.

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

1. **Provide the SAS URI**:
   * Paste the **Blob SAS URL** copied earlier into the required field.
   * Enter a display name for the connection and click **Next**.

A screenshot of a computer

Description automatically generated

1. **Verify and Connect**:
   * Follow the prompts to complete the connection. Once connected, the ADLS container will appear in the explorer.

A screenshot of a computer

Description automatically generated

**Step 7: Upload Files to ADLS**

1. **Navigate to the Container**:
   * Locate your ADLS container in Azure Storage Explorer.
2. **Upload Files**:
   * Right-click on the container and choose **Upload Files** or **Upload Folder**.
   * Select the files/folders to upload from your local machine.

A screenshot of a computer

Description automatically generated

**Step 8: Verify Connection and File Upload**

1. Open the container in Azure Storage Explorer to confirm that the files are uploaded successfully.
2. You can now manage the ADLS files using the tool.

A screenshot of a computer

Description automatically generated

**Run the notebook of delta live table taxi case study**

Get the New York City Taxi Trip Duration dataset from Kaggle

**1. Creating the taxi\_raw\_records Streaming Table**

%sql

CREATE OR REFRESH STREAMING TABLE taxi\_raw\_records

AS

SELECT \*

FROM

STREAM(samples.nyctaxi.trips);

A screenshot of a computer

Description automatically generated

**Description:**

* This query creates a **streaming table** named taxi\_raw\_records.
* It continuously ingests data from the samples.nyctaxi.trips stream.
* The table is refreshed automatically as new records arrive, ensuring real-time updates.

**2. Creating the taxi\_raw\_recordsonadb Streaming Table with Data Quality Constraints**

%sql

CREATE OR REFRESH STREAMING TABLE taxi\_raw\_recordsonadb

(CONSTRAINT valid\_distance EXPECT (trip\_distance > 0.0) ON VIOLATION DROP ROW )

AS SELECT

  \*

FROM

  STREAM(samples.nyctaxi.trips);

A screenshot of a computer

Description automatically generated

**Description:**

* This query creates another **streaming table** named taxi\_raw\_recordsonadb with a **data quality constraint**.
* It only includes records where trip\_distance > 0.0. Rows violating this condition are **dropped**.
* The table pulls data from the samples.nyctaxi.trips stream and ensures that only valid rows are stored.

**3. Creating the flagged\_rides Streaming Table**

%sql

CREATE OR REFRESH STREAMING TABLE flagged\_rides

AS SELECT

  date\_trunc('week', tpep\_pickup\_datetime) as week,

  pickup\_zip as zip,

  trip\_distance, fare\_amount

FROM

 STREAM(samples.nyctaxi.trips)

WHERE   ((pickup\_zip = dropoff\_zip AND fare\_amount > 50) OR

        (trip\_distance < 5 AND fare\_amount > 50));

A screenshot of a computer

Description automatically generated

**Description:**

* This query creates a **streaming table** named flagged\_rides for identifying specific ride patterns.
* It filters rides with the following conditions:
  + The pickup and dropoff locations are the same, and the fare exceeds $50.
  + The trip distance is less than 5 miles, but the fare exceeds $50.
* The output includes truncated weekly pickup times, pickup zip codes, trip distances, and fare amounts.

**4. Creating the weekly\_stats Materialized View**

%sql

CREATE

OR REFRESH MATERIALIZED VIEW weekly\_stats

AS SELECT

  date\_trunc("week", tpep\_pickup\_datetime) as week,

  AVG(fare\_amount) as avg\_amount,

  AVG(trip\_distance) as avg\_distance

FROM

 live.taxi\_raw\_records

GROUP BY

  week

ORDER by week ASC;

A screenshot of a computer

Description automatically generated

**Description:**

* This query creates a **materialized view** named weekly\_stats to compute aggregated statistics.
* It calculates the following metrics for each week:
  + Average fare amount (avg\_amount).
  + Average trip distance (avg\_distance).
* Data is grouped by truncated weekly pickup times (week) and sorted in ascending order by week.

**5. Creating the top\_m Materialized View**

%sql

CREATE OR REPLACE MATERIALIZED VIEW top\_m

AS SELECT

  weekly\_stats.week,

  ROUND(avg\_amount,2) as avg\_amount,

  ROUND(avg\_distance,3) as avg\_distance,

  fare\_amount,trip\_distance, zip

  FROM live.flagged\_rides

  LEFT JOIN live.weekly\_stats ON flagged\_rides.week = weekly\_stats.week

  ORDER BY fare\_amount DESC;

A screenshot of a computer

Description automatically generated

**Description:**

* This query creates or replaces a **materialized view** named top\_m for detailed analysis.
* It combines data from flagged\_rides and weekly\_stats:
  + Uses a **left join** to associate weekly flagged rides with their corresponding statistics.
  + Includes weekly averages (avg\_amount, avg\_distance), fare amounts, trip distances, and zip codes.
* The result is sorted by fare\_amount in descending order, showing the most expensive rides first.