Data Pipeline Documentation for Trigger Usage

This document outlines the steps and trigger selection for moving data from an HTTP API and Microsoft SQL Server (SSMS) to a storage account, processing the data in Azure Databricks, and transferring it to processed and staging containers. Each step includes the trigger used, how it was implemented, and the rationale behind the choice.

Step 1: Data Movement from Source to Raw Container

Objective:

- Copy data from HTTP API to the raw container.
- Copy data from Microsoft SQL Server (SSMS) to the raw container.

Tools Used:

- Azure Data Factory (ADF).

Pipeline Design:

- A single pipeline (pipeline_casestudy) containing two **Copy Data** activities:
    1. **HTTP API to Raw Container.**
    2. **SSMS to Raw Container.**

Trigger Used: Scheduled Trigger

- **How:**
    - Configured a **Scheduled Trigger** in Azure Data Factory to run the pipeline every hour (or based on data refresh requirements).
    - The schedule was set for regular intervals (e.g., every 6 hours) using the Azure Data Factory trigger interface.
- **Why:**
    - HTTP API and SSMS data are expected to update at regular intervals.

    - Scheduled Trigger ensures both data sources are captured at the same time, simplifying downstream processing.

Step 2: Data Cleaning in Azure Databricks

Objective:

- Import data from the raw container into Azure Databricks.
- Clean and transform data (e.g., remove duplicates, handle null values, and filter rows).
- Save the cleaned data to the processed container using **append**.

Trigger Used: Event-Based Trigger

- **How:**
    - Configured an **Event-Based Trigger** in Azure Data Factory to monitor the raw container for new files.
    - The trigger launches a Databricks notebook via Azure Data Factory's **Notebook Activity** whenever new data arrives in the raw container.
- **Why:**
    - Automates the data cleaning process immediately after new data is copied to the raw container.
    - Ensures the processed container always contains the latest cleaned data without requiring manual intervention.

Step 3: Joining Processed Data and Moving to Staging Container

Objective:

- Join cleaned data from the processed container (HTTP API and SSMS files).
- Save the joined data to the staging container.

Tools Used:

- Azure Databricks for joining data.
- Azure Data Factory for staging data movement.

Trigger Used: Scheduled Trigger

- **How:**
  - Scheduled a trigger to run the pipeline that moves joined data from the processed container to the staging container daily at midnight.
  - The pipeline launches a Databricks notebook to join the data, then writes the result to the staging container using a **Copy Data Activity** in Azure Data Factory.
- **Why:**
  - Staging data is typically used for downstream reporting or integration, which often requires updates at fixed intervals.
  - Using a Scheduled Trigger ensures predictable updates to the staging container.

Detailed Steps

1. **Copy Data from Sources to Raw Container:**
   - **HTTP API:**
     - Use a **Copy Data Activity** with a REST linked service.
     - Configure source to the API endpoint (https://jsonplaceholder.typicode.com/users).
     - Configure the sink to write the data to the raw container in Parquet format.
   - **SSMS:**
     - Use a **Copy Data Activity** with a self-hosted integration runtime linked service.
     - Configure source to Microsoft SQL Server (e.g., dbo.football table).

        - Configure the sink to write the data to the raw container in CSV format.

2. **Event-Based Trigger for Databricks Cleaning:**
    - Set up an **Event-Based Trigger** to monitor the raw container.
    - Launch an Azure Databricks notebook using an **ADF Notebook Activity**.
    - Databricks Notebook:
        - Import raw files (Parquet and CSV).
        - Clean data:
            - Filter rows, remove duplicates, handle nulls, etc.
        - Save cleaned data to the processed container using append.

3. **Scheduled Trigger for Processed to Staging:**
    - Set up a **Scheduled Trigger** in Azure Data Factory to run the pipeline at midnight daily.
    - Azure Databricks Notebook:
        - Read cleaned data from the processed container.
        - Perform the join operation:
            - Match rows using common fields (e.g., user_id).
        - Save joined data to the staging container.
    - Configure the sink in the pipeline to write to the staging container in CSV format.

Trigger Summary

Stage	Trigger Type	Reason for Choice
Raw Data Ingestion	Scheduled Trigger	HTTP API and SSMS data updates at regular intervals; ensures predictable ingestion times.
Cleaning in Databricks	Event-Based Trigger	Automates cleaning whenever new data is available in the raw container.
Processed to Staging	Scheduled Trigger	Provides regular updates to the staging container for downstream usage.

Key Notes:

- **Secret Management:** Used Azure Key Vault for secure connection strings and credentials in both ADF and Databricks.
- **Version Control:** Linked ADF to a GitHub repository (DataLake) and performed all changes in the dev branch, followed by a pull request to the QA branch.
- **File Formats:**
    - Parquet for HTTP API data (efficient for analytics).
    - CSV for SSMS data (widely compatible).

This document ensures a clear understanding of the triggers, their implementations, and the reasons behind their selection for this pipeline.