### **Data set: https://www.kaggle.com/c/nyc-taxi-trip-duration/data**

### 

### **Problem Statement**

The goal of the ETL task is to process and prepare the 2016 NYC Yellow Cab trip record dataset for predictive analysis using Azure Data Lake and Databricks Delta Lake. The dataset comprises trip attributes, and the objective is to create a robust ETL pipeline based on the **Bronze, Silver, and Gold architecture** to enable efficient and scalable data transformation and preparation.

The final outcome of this ETL pipeline is a **Gold layer dataset** that can be directly used for predicting trip durations in the test set while maintaining data integrity, accuracy, and compliance with best practices.

### **Solution Architecture**

**Components:**

1. **Azure Data Lake Storage (ADLS)**: Serves as the primary data lake for raw and processed data.
2. **Azure Databricks**: Provides a collaborative environment for processing and transforming the data.
3. **Databricks Delta Lake**: Ensures reliability, consistency, and versioning of the data with ACID transactions.
4. **Bronze, Silver, Gold Architecture**:
   * **Bronze Layer**: Stores raw, unprocessed data as ingested from the source.
   * **Silver Layer**: Holds cleaned, enriched, and transformed data for analytical and processing use cases.
   * **Gold Layer**: Contains aggregated, highly curated datasets ready for consumption by machine learning models or reporting tools.

### **Detailed Steps for the ETL Pipeline**

#### **1. Ingestion Phase: Populate the Bronze Layer**

* **Data Source**:
  + The dataset (train.csv, test.csv, sample\_submission.csv) will be uploaded to the **Bronze layer** in Azure Data Lake.
* **Process**:
  + Use Databricks notebooks to load data from Azure Data Lake into Delta tables.
  + Store the data as-is in Delta format for raw archival.

**Tasks:**

1. Mount Azure Data Lake to Databricks using service principal or Azure AD authentication.
2. Ingest the dataset files into the **Bronze Delta Lake tables**.

**Schema:**

* train\_bronze: Raw trip records (from train.csv).
* test\_bronze: Raw trip records (from test.csv).

#### **2. Transformation Phase: Populate the Silver Layer**

* **Objective**: Clean and enrich the data to handle missing, inconsistent, or invalid values.
* **Transformations**:
  + **Datetime Parsing**:
    - Convert pickup\_datetime and dropoff\_datetime to standard datetime format.
    - Calculate the trip\_duration as the difference between dropoff\_datetime and pickup\_datetime.
  + **Filter Invalid Records**:
    - Remove trips with pickup\_datetime greater than dropoff\_datetime.
    - Remove trips with unreasonable durations or distances.
  + **Coordinate Validation**:
    - Ensure longitude and latitude values fall within valid ranges for NYC.
  + **Derived Features**:
    - Add new columns like day\_of\_week, hour\_of\_day, and trip\_distance (using Haversine formula).
  + **Data Deduplication**:
    - Eliminate duplicate records based on id.
* **Output**:
  + A cleaned and enriched Delta Lake table for the training and testing datasets.

**Tasks:**

1. Read data from the Bronze layer.
2. Perform transformations using PySpark in Databricks.
3. Write the output to **Silver Delta Lake tables**.

**Schema**:

* train\_silver: Cleaned and enriched training data.
* test\_silver: Cleaned and enriched test data.

#### **3. Aggregation Phase: Populate the Gold Layer**

* **Objective**: Aggregate and prepare the data for model consumption.
* **Transformations**:
  + Feature engineering:
    - Generate statistical summaries like average trip duration and distance by vendor\_id, day\_of\_week, and hour\_of\_day.
    - Create predictive features such as traffic\_intensity or rush\_hour indicators.
  + Partition and Optimize:
    - Partition data by vendor\_id or day\_of\_week for efficient querying.
    - Optimize Gold layer tables using Delta Lake OPTIMIZE and ZORDER by critical columns.
* **Output**:
  + Final dataset for training a machine learning model and making predictions.

**Tasks:**

1. Read from Silver layer tables.
2. Perform aggregations and create derived features.
3. Store the results in the **Gold Delta Lake tables**.

**Schema**:

* train\_gold: Prepared dataset for ML model training.
* test\_gold: Prepared dataset for ML model predictions.