**ETL Pipeline Documentation for NYC Restaurant Inspection Data**

**1. Data Understanding**

The dataset is sourced from the NYC Open Data Portal and contains restaurant inspection records. Key characteristics include:

* **Source:** [NYC Restaurant Inspection Data](https://data.cityofnewyork.us/Health/DOHMH-New-York-City-Restaurant-Inspection-Results/43nn-pn8j/about_data)
* **Key Columns:**
  + **CAMIS:** Unique restaurant ID
  + **DBA:** Restaurant name
  + **BORO:** Borough (e.g., Manhattan, Queens)
  + **ZIPCODE:** Location ZIP code
  + **CUISINE DESCRIPTION:** Type of cuisine
  + **INSPECTION DATE:** Date of inspection
  + **ACTION:** Inspection actions taken
  + **VIOLATION CODE & DESCRIPTION:** Violation details
  + **CRITICAL FLAG, SCORE, GRADE, GRADE DATE, INSPECTION TYPE, LATITUDE, LONGITUDE, RECORD DATE:** Additional details

**2. Data Model**

The data model is structured as follows:

* **Primary Key:** CAMIS uniquely identifies each restaurant.
* **Attributes:** Include various data types (text, timestamps, integers, floats).
* **Partitioning:**
  + Data is partitioned by the BORO column, enabling efficient query performance.
* **Storage Format:**
  + Data is stored in CSV format (with the option to use Delta format for additional benefits such as ACID transactions) in Azure Data Lake Storage (ADLS) via Databricks.

**3. Solution Architecture**

The pipeline’s solution architecture comprises the following layers:

* **Data Ingestion:**
  + Reads raw CSV data from ADLS.
* **ETL Pipeline:**
  + **Extraction:** Loads raw data from ADLS.
  + **Transformation:**
    - Converts data types (e.g., date and numeric fields).
    - Cleans and standardizes data (trims text, imputes missing values).
    - Filters incremental data using a watermark (latest processed INSPECTION DATE).
  + **Loading:**
    - Writes the transformed data back to ADLS, partitioned by BORO.
* **Watermark Management:**
  + Stores the latest inspection date to ensure only new records are processed in subsequent runs.
* **Orchestration:**
  + The entire process is scheduled to run daily as a Databricks job or workflow.

**Architecture Diagram:**

csharp

CopyEdit

[Raw Data in ADLS]

│

▼

[Databricks ETL Pipeline]

│

▼

[Transformed & Partitioned Data in ADLS]

│

▼

[Updated Watermark for Incremental Loads]

**4. ETL Pipeline Screenshots**

Below are key code snippets representing steps in the ETL process. (Replace these code blocks with actual screenshots from your Databricks notebook if needed.)

**a. Data Loading & Incremental Filtering**

python

CopyEdit

# Load raw CSV data

df = spark.read.option("header", "true").csv(raw\_path)

# Convert inspection date to timestamp and filter by last processed date

df = df.withColumn("INSPECTION DATE", to\_timestamp(col("INSPECTION DATE"), "MM/dd/yyyy"))

df = df.filter(col("INSPECTION DATE") > last\_inspection\_date)

**b. Data Cleaning & Transformation**

python

CopyEdit

# Drop unnecessary columns

columns\_to\_drop = ["BUILDING", "STREET", "PHONE", "COMMUNITY BOARD",

"COUNCIL DISTRICT", "CENSUS TRACT", "BIN", "BBL", "NTA"]

df\_cleaned = df.drop(\*columns\_to\_drop)

# Convert data types

df\_cleaned = df\_cleaned.withColumn("SCORE", col("SCORE").cast(IntegerType()))

**c. Writing Transformed Data**

python

CopyEdit

# Write transformed data partitioned by BORO

df\_cleaned.write \

.mode("append") \

.partitionBy("BORO") \

.option("header", "true") \

.csv(transformed\_path)

**d. Updating the Watermark**

python

CopyEdit

# Capture the latest inspection date from the cleaned data

max\_date = df\_cleaned.agg({"INSPECTION DATE": "max"}).collect()[0][0]

# Update watermark for incremental loading

watermark\_df = spark.createDataFrame([(str(max\_date),)], ["max\_inspection\_date"])

watermark\_df.write.mode("overwrite").option("header", "true").csv(watermark\_path)

Datapipeline:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A computer screen shot of text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A close-up of a computer screen

AI-generated content may be incorrect.

Raw Data:

A screenshot of a mail

AI-generated content may be incorrect.

Transformed Data:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.