AWS + Databricks (Problem Statement)

Problem Statement:

XYZ is a data analytics firm specializing in handling the Taxi Ride dataset provided by clients. As a data analyst at XYZ, your primary task involves executing an Extract, Transform, Load (ETL) job to meticulously reshape the data according to the specific requirements outlined by the client.

Use the following steps to get the work done.

Phase 1: Ingestion

1] Create a S3 bucket by the name tlc-trip-record-data-lake-<yourname-random number>

[ Ex: tlc-trip-record-data-lake-tina-1001]

2] Create 3 folders under the same bucket by the name “raw-layer”, “processed-layer” and “reference-layer”.

3] Load raw data files for Q2-2020 in the raw-layer with the following hierarchy as

raw/2020/Q2 and lookup file in reference-layer, respectively.

Phase 2: Cataloging

1] Create a crawler for raw-layer as crawler-raw-layer-<yourname>

* Select S3 as data source and choose path as s3://tlc-trip-record-data-lake-<yourname>random number>/raw-layer/2020/Q2/
* Database name: tlc-trip-record-data-db- <yourname>

3] Create the crawler for reference layer as crawler-reference-layer-<yourname>

* Database name: tlc-trip-record-data-db-<yourname> [ the same one as above]

Phase-3: Ad-Hoc Exploration – SQL Analytics

1] Go to your S3 bucket and create a new folder named as “results”.

2] Go to Athena>>Query Editor>>Settings>> Manage and set the Query Result location

and encryption as s3://tlc-trip-record-data-<yourname-randomnumber>/results and

save

3] Perform ad-hoc SQL Analytics on raw table and reference table.

4] Now let’s get rid of these double quotes you saw in last query execution for reference data.

Go to AWS Glue >> Tables >> your reference layer table >> Actions >> Edit table >>

Under Serialization lib: replace the value with

org.apache.hadoop.hive.serde2.OpenCSVSerde And update the Serde parameters as follows:

|  |  |
| --- | --- |
| Key | Value |
| escapeChar | \ |
| quoteChar | “ |
| separatorChar | , |

5] Now go back to Athena and query your reference table and see the results

Phase-4: Processing

A] Let’s Transform Taxi Ride Data

1] Let’s add transformation for dropping null fields

2] Add next transform, actions >> ApplyMapping

* Transform: Rename the columns as: vendor\_id, pickup\_datetime, dropoff\_datetime and change the datatype for pickup\_datetime and dropoff\_datetime to timestamp
* Drop ratecodeid and store\_and\_fwd\_flag columns
* Rename: pulocationid to pickup\_locationid and dolocationid to drop\_off\_locationid
* Drop the following columns :

fare\_amount, extra, mta\_tax, tip\_amount, tolls\_amount, improvement\_surcharge, congestion\_surcharge

3] Add next transformation, actions>>Filter for VendorId

* Transform: Choose Global OR and in filter condition add filter condition as mentioned.

|  |  |  |
| --- | --- | --- |
| key | operation | Value |
| Vendor\_id | = | 1 |
| Vendor\_id | = | 2 |

4] Add next transform. Actions >> Filter for Filter Passenger Count

* Transform: Select Global AND and add filter condition as shown below.

|  |  |  |
| --- | --- | --- |
| key | operation | Value |
| Passenger\_count | > | 0 |

**B] Let’s Transform** Pick Up - Reference Data

1] Add next transform, actions >> ApplyMapping

* Node Name: Pick Up Change Schema
* Transform: Rename the columns as follows: Append pick\_ before every name in Target key

3] Add join transform where you will join Filter Passenger Count and Pick Up Change Schema

* Node Name: Filtered Taxi Data + P
* Join type: Inner Join

|  |  |
| --- | --- |
| Pick up Change Schema | Filter Passenger Count |
| Pick\_locationid = | Pickup\_location\_id |

**C]**

**1]** Click in blank space on canvas and one more S3 source.

* Node Name: Drop Reference
* Data source: Select Data Catalog table as S3 source type. Select Glue Database created earlier and select reference table again.

2] Add ApplyMapping transform to this new source.

* Node Name: Drop Change Schema
* Transform: Simply append drop\_ for each column in target key

3]Click in the blank space on the canvas. Add one more Join transform.

* Node Name: Join Filtered Taxi + P + D and select Drop Change Schema and Filtered Taxi Data + P as node parents.
* Join type: Inner Join

|  |  |
| --- | --- |
| Filtered Taxi Data + P | Drop Change Schema |
| Drop\_off\_locationid = | drop\_location\_id |

4]Now after final joined node i.e. Join Filtered Taxi + P + D, add ApplyMapping and drop these two   
pickup\_locationid and drop\_off\_locationid coloumns.

5] Last step, add target. Select S3

* Node Name: Write back to S3
* Format -> Parquet
* Compression -> Snappy
* target path as s3://tlc-trip-record- data-lake-mi-/processed-layer/

D] Edit job details.

1] Job Name: Transforming Taxi Ride Data and select GlueCapstoneRole

2] IAM Role

3] Requested number of workers: 2

4] Give a unique name to your script. <unique name>.py

5] Uncheck Job metrics, Continuous logging and Spark UI

E]

1] Save the job and run

2] Watch for the output in processed layer in S3

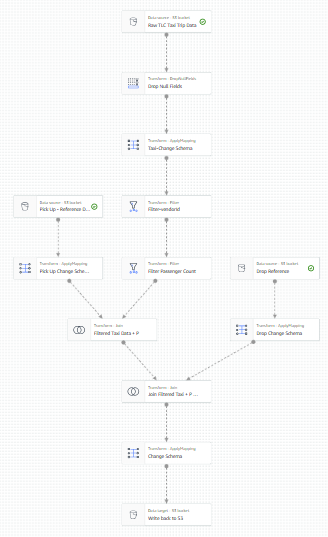
3] Create a crawler for processed layer. Name as crawler-processed-layer-<yourname>

4] Choose S3 as data source.

Provide s3://tlc-trip-record-data-lake-<yourname-randomnumber>/processed-layer/ as S3 path

5] Select your existing Glue database as target database and create the crawler. Run it.

6] Perform SQL Analytics using Athena on processed layer.

****

Kindly take the screenshots at each and every step of your implementation in order to complete your final submission in a PDF format.