Code Assessment Document (v1.1)

By

Chitwan Humad

Contents

Assessment Details

Problem Statement:

- Create an ELT pipeline that ingests a CSV dataset (choose any sufficiently dense source eg. https://www.kaggle.com/datasets/abdullah0a/telecom-customer-churn-insights-for-analysis).
- Load up the dataset into a staging database of your choice.
- Design a transformation layer to process the input dataset for missing values (use defaults) and anonymising PII.
- The destination for the processed data should be a database ideal for generating reports.
- Establish an orchestration workflow for this pipeline to accept a feed every hour (should be configurable).
- Integrate any open-source reporting tool to generate statistics about the flow.
- Ensure the entire setup is available through composable container definition(s).

Tech Stack:

- Language/frameworks/solutions of your choice. Please just ensure, the solution is easy to run on a laptop.
- Please use open-source solutions wherever possible.

Delivery:

- Please share the entire source code as a public Github repository.
- Do add relevant instructions to run the code.
- Please also ensure it stays accessible for the duration of the discussions with HGI.

ETA:

- Please ensure the assignment is completed in about 16-20 hours (can be split over days if practical schedules demand).

Assessment – In scope:

- Bronze Layer (Data Lake): Read and writing csv data via a pipeline to store in the database table – csv data will be pulled up from Kaggle
- Silver Layer (Transformed Data layer): Data transformation and stored the refined data into database tables – up to 3 use cases
- Gold Layer (Reporting Layer): Pre-aggregated data for reporting purposes – up to 3 use cases

Assessment – Out of scope:

- Containerization of the solution

Assessment – Use Case Success Criteria:

The solution should be considered as successful if the following use cases are achieve during the user acceptance testing:

1. Tech stack selection: should be Open source as far as possible
2. Each run should have internal runid to track pipeline runs
3. CSV file(s) should be able read from <>/in/<name>.csv folder and load into the database without any change in the data in the raw_customer table of bronze_db database
4. Solution to enable hourly to ingest a new file hourly
5. The processes file should be moved/archived in the processed file into <>/processed/<name>_runid _datetimeid _done.csv
6. Solution should follow 3 use cases to conduct data transformations:
    a. Check for NaN or missing values for a few fields (field names – TBD)
    b. Check valid values for Age – should be a positive integer only

    c. Check for valid values from the data dictionary for ContractType field as Month-to-Month, One-Year, Two-Year
7. Bad data rows based on the above should be saved into the <>/baddata/<name>_runid _datetimeid _done.csv
8. Read bronze_db.raw_customer table data and perform following transformation to make presentable reports:
    a. Transform InternetService missing values to None
    b. Round off TotalCharges values to 2 decimals
    c. Define new dimension as Tenure_Range for each 10 blocks, e.g. 1-10, 11-20 so on
    d. Define Age_band dimension 20-25, 36-30 so on every 5 years
    e. Drop Age field to preserve PII information
    f. Define new dimension Category High/Medium/Low for MonthlyCharges < 50 Low, 51-100 medium and > 100 high
9. The transformed data should be stored into silver_db.customer table
10. Produce a aggregated data models to generate various reports in the gold_db.<table_names>, like:
    a. Count of customers by Categories (i.e. High/Medium/Low)
    b. Aggregated revenue (TotalCharges) by Contract Types
    c. Aggregated revenue (TotalCharges) by InternetService
    d. Customer demographic Presentation who availed technical support facility by Age_band and gender
11. A run and log table to record runs

Assessment – Tech Stack Selection: Tentative

- OS – Windows laptop
- Prefect for data pipeline – Open source
- Superset or Birt – Open source
- Database – Sqlserver Express using sa credentials

Assessment – Deliverables:

- Git repo url https://github.com/chitwanhumad/hg_datapipeline

(Kindly confirm you can access the url)

Assessment – Completion Date: Tentative

22-Aug-2025

Solution High Level

1. SCD 2 Implementation – maintained all history however report shows all latest data for each customer. Example –

Runid	New CustomerID in the input file	Updated CustomerID in the input file	Report Data
1	1- 100	NA	All 1 - 100
2	101 - 120	5, 50	All 1 – 120**
3	NA	61, 71	All 1 – 120**

** updated records data with the latest rows

2. No archival of old data has been provisioned however it has to be there. Suggested solution could be, based on the business requirements last N days data should be kept into silver_db.customers tables as per business policy. The system performance will degrade without data archival policies.
3. Only one condition of Bad data has been assumed for now. It is for non int customerID.
4. It is assumed that there could be more than one customer files may be loaded at a time. All good data and/or bad data will be saved in the /archive/ folder with the runid in the file.

5. Runid is to track every run. The same runid will be used to read logs from the dbo.acr_log table.
6. Runid will also be used for data lineage purpose.
7. Bronze_db will have all data, each row will have a runid and inserttime associated with it.
8. Siliver db will have soft delete of the older rows, referece column is is_current = 'Y'
9. Gold_db will have up-to-date aggregated data only. Users can refer lastrefreshtime field for their reference.

Solution Installation Steps

1. Make sue you have python 3.10 environment
2. Run steps as per Environment.txt
3. Sync the git repo in your windows laptop
   https://github.com/chitwanhumad/hg_datapipeline
4. Install packages from requirements.txt
   pip install -r requirements.txt
5. Open config.ini file to update your sql server database credentials. Update server name, user and password.
6. Configure your root directory where the source file will be placed.
7. Run \ddl\dbsetup.sql using SQL Server management studio. This script will create all required databases, tables and other db objects.

Set up and Run Data pipeline

1. Copy the source file into the <>/source/*.csv folder, there could be more than 1 incoming files.
2. **Run /workflow/main.py**
3. Check Run status and logs in by using following queries

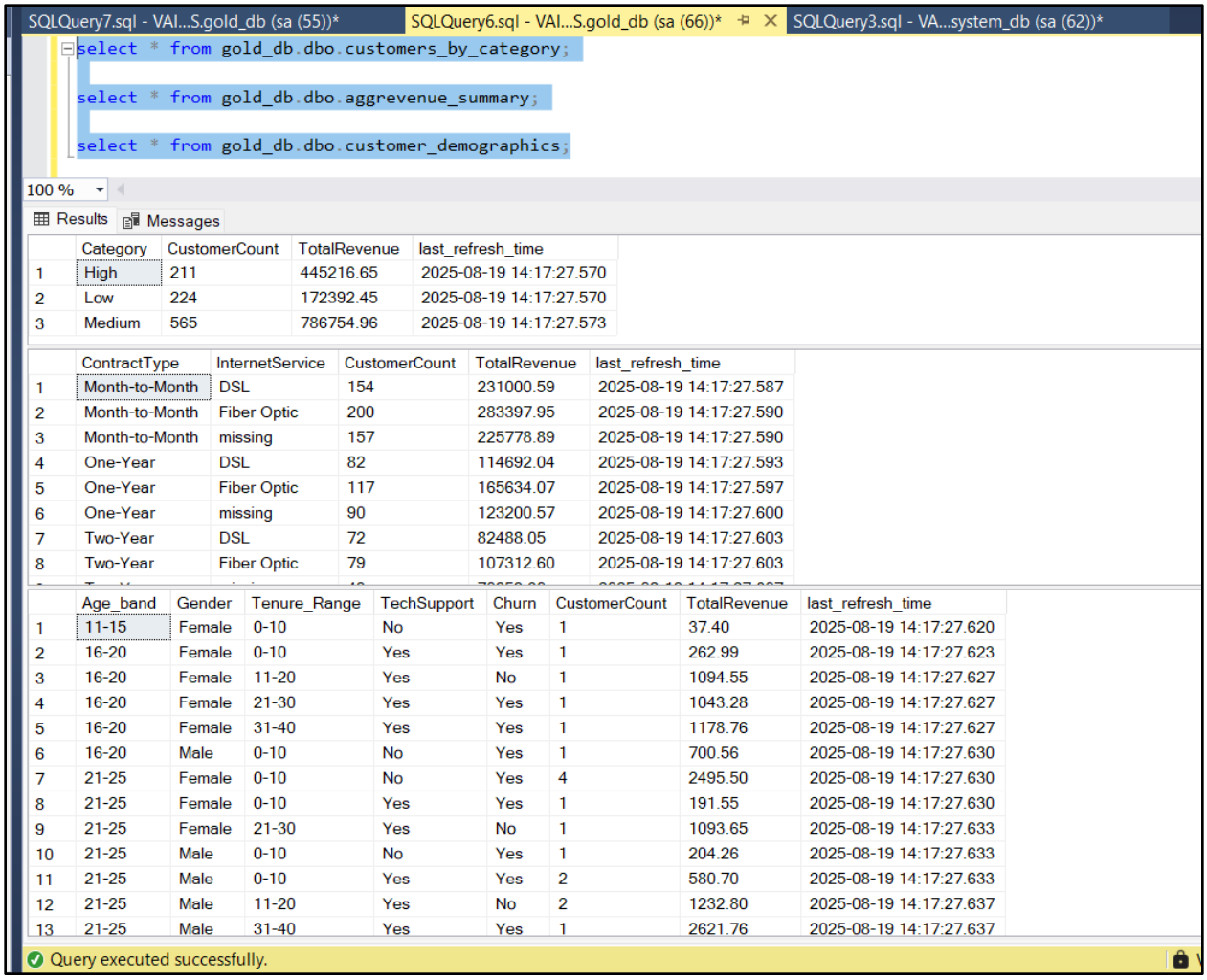
```
select * from system_db.dbo.ach_runs where runid = ?;

select * from system_db.dbo.ach_logs where runid = ?;
```

4. If now ERROR in the above step, verify your data in bronze and silver layers databases by using following queries

```
select * from bronze_db.dbo.raw_customers where runid = 51

select * from silver_db.dbo.customers where is_current = 'Y' order by CustomerID;
```

5. To view the modeled data for reports, run following sqls:

```
select * from gold_db.dbo.customers_by_category;

select * from gold_db.dbo.aggrevenue_summary;

select * from gold_db.dbo.customer_demographics
```



	Category	CustomerCount	TotalRevenue	last_refresh_time
1	High	211	445216.65	2025-08-19 14:17:27.570
2	Low	224	172392.45	2025-08-19 14:17:27.570
3	Medium	565	786754.96	2025-08-19 14:17:27.573

	ContractType	InternetService	CustomerCount	TotalRevenue	last_refresh_time
1	Month-to-Month	DSL	154	231000.59	2025-08-19 14:17:27.587
2	Month-to-Month	Fiber Optic	200	283397.95	2025-08-19 14:17:27.590
3	Month-to-Month	missing	157	225778.89	2025-08-19 14:17:27.590
4	One-Year	DSL	82	114692.04	2025-08-19 14:17:27.593
5	One-Year	Fiber Optic	117	165634.07	2025-08-19 14:17:27.597
6	One-Year	missing	90	123200.57	2025-08-19 14:17:27.600
7	Two-Year	DSL	72	82488.05	2025-08-19 14:17:27.603
8	Two-Year	Fiber Optic	79	107312.60	2025-08-19 14:17:27.603

	Age_band	Gender	Tenure_Range	TechSupport	Churn	CustomerCount	TotalRevenue	last_refresh_time
1	11-15	Female	0-10	No	Yes	1	37.40	2025-08-19 14:17:27.620
2	16-20	Female	0-10	Yes	Yes	1	262.99	2025-08-19 14:17:27.623
3	16-20	Female	11-20	Yes	No	1	1094.55	2025-08-19 14:17:27.627
4	16-20	Female	21-30	Yes	Yes	1	1043.28	2025-08-19 14:17:27.627
5	16-20	Female	31-40	Yes	Yes	1	1178.76	2025-08-19 14:17:27.627
6	16-20	Male	0-10	No	Yes	1	700.56	2025-08-19 14:17:27.630
7	21-25	Female	0-10	No	Yes	4	2495.50	2025-08-19 14:17:27.630
8	21-25	Female	0-10	Yes	Yes	1	191.55	2025-08-19 14:17:27.630
9	21-25	Female	21-30	Yes	No	1	1093.65	2025-08-19 14:17:27.633
10	21-25	Male	0-10	No	Yes	1	204.26	2025-08-19 14:17:27.633
11	21-25	Male	0-10	Yes	Yes	2	580.70	2025-08-19 14:17:27.633
12	21-25	Male	11-20	Yes	No	2	1232.80	2025-08-19 14:17:27.637
13	21-25	Male	31-40	Yes	Yes	1	2621.76	2025-08-19 14:17:27.637

Set up Local Superset Environment

1. Download install Superset
   a. Python virtual environment

      python -m pip install --upgrade pip setuptools wheel

      python -m venv venv

      venv\Scripts\activate
      (every time when you start server)

   b. Install apache-superset

      pip install apache-superset

   c. create a new secret

      python -c "import secrets; print(secrets.token_urlsafe(64))"

   d. save this secret inside into **D:\superset\superset_config.py**

      import os

      SECRET_KEY = "my_random_long_secret_key_123!@#"

      Note: example superset_config.py file canbe referred from

      \superset\superset_config.py

   e. Make sure you have the below package

      pip install marshmallow==3.20.1

      pip install pymssql # sql server connector

      pip install sqlalchemy==2.0.25

      pip install pyodbc

      pip install --upgrade apache-superset

   f. Set up variables and flask application

      set SUPERSET_CONFIG_PATH=D:\superset\superset_config.py

      set FLASK_APP=superset.app:create_app()

      (every time when you start server)

   g. run db upgrade command

      superset db upgrade

(one time only)

h. To create admin user and complete the prompt for u/p

superset fab create-admin
(one time only)

i. Load examples

superset load_examples
(one time only)

j. Initialize superset

superset init
(every time when you start server)

k. Start server
superset run -h localhost -p 8088
(every time when you start server)

l. Access server
http://localhost:8088

Start Superset Server:

1. Refer file \superset\start_superset_server.bat
2. Set up path where your python virtual environment for superset is available
3. Run the batch file

   \superset\start_superset_server.bat

4. Break it to stop server

Create DataSource

1. Create datasource for gold_db

   mssql+pymssql://sa:unica*03@VAIBHAVI:52557/gold_db

NOTE: In my case the instance name is SQLEXPRESS, it's dynamic port is 52557 so used dynamic port, check TCP/IP settings to fetch the same.

2. Test connection, save dataspurce name as **GoldDB**

Import Dashboards

1.