

Data Engineer Project - Dibimbang

# CREDIT DATA PIPELINE

Muhammad Arij Arfina



# Content

01

Problems

02

Data Pipeline

03

Requirements & Steps

04

Output



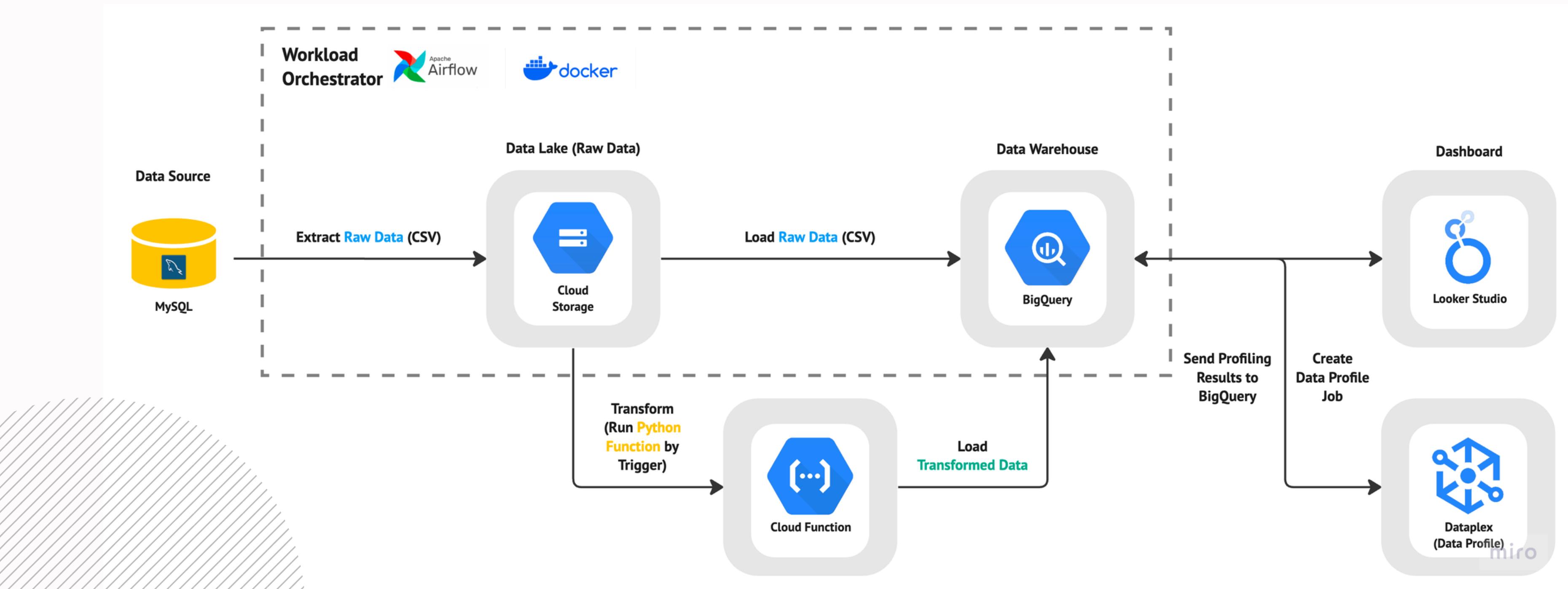


# Problems

Credivo, like many modern companies, deals with a lot of data from different sources. This data can help make smart decisions and find important insights, but it's hard to use effectively.

To fix this, Credivo knows it needs a better data pipeline for managing data. This pipeline will organize the data better, making it easier to use and trust. With this pipeline, Credivo wants to give its team the right information at the right time, so they can make better decisions and make the company better overall.

# Data Pipeline



# Requirement

## 1. Data Credit:

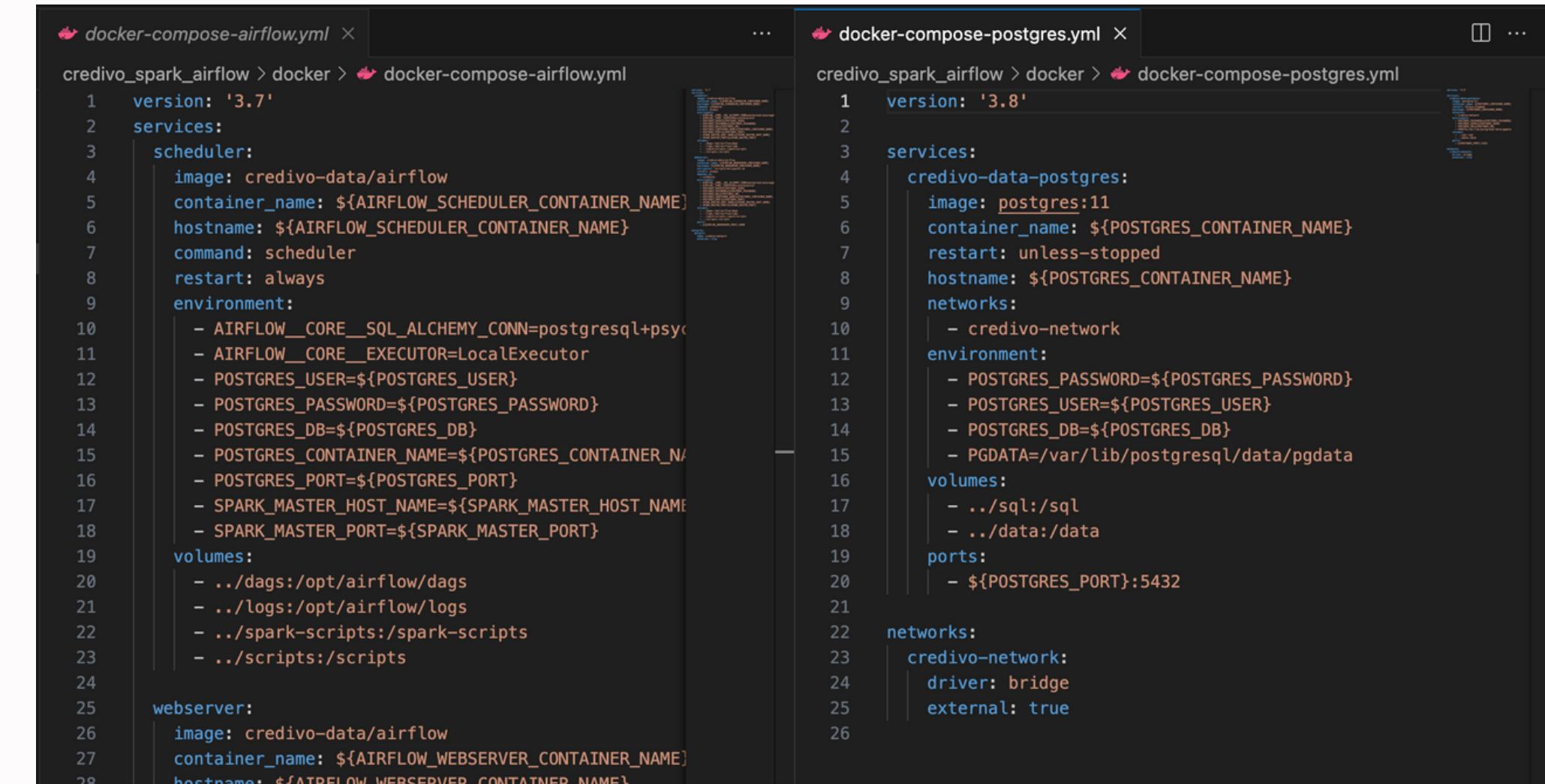
- a. Total Column : 121 Column
- b. Total Row : 48.744

Column Name	#	Data Type
123 SK_ID_CURR	1	int
RBC NAME_CONTRACT_TYPE	2	varchar(50)
RBC CODE_GENDER	3	varchar(50)
RBC FLAG_OWN_CAR	4	varchar(50)
RBC FLAG_OWN_REALTY	5	varchar(50)
123 CNT_CHILDREN	6	int
123 AMT_INCOME_TOTAL	7	double
123 AMT_CREDIT	8	double
123 AMT_ANNUITY	9	double
123 AMT_GOODS_PRICE	10	double
RBC NAME_TYPE_SUITE	11	varchar(50)
RBC NAME_INCOME_TYPE	12	varchar(50)
RBC NAME_EDUCATION_TYPE	13	varchar(50)
RBC NAME_FAMILY_STATUS	14	varchar(50)
RBC NAME_HOUSING_TYPE	15	varchar(50)
123 REGION_POPULATION_RELATIVE	16	double
123 DAYS_BIRTH	17	int
123 DAYS_EMPLOYED	18	int
123 DAYS_REGISTRATION	19	double
123 DAYS_ID_PUBLISH	20	int
123 OWN_CAR_AGE	21	double
123 FLAG_MOBIL	22	int

# Requirement

## 2. Docker Compose Configuration:

- File:
  - **docker-compose-airflow.yml**
  - **docker-compose-postgres.yml**
  - **docker-compose-jupyter.yml**
- Services:
  - Airflow Scheduler
  - Airflow Webserver
  - Postgres
  - Jupyter



The image shows a code editor with two tabs open, each displaying a Docker Compose configuration file.

**docker-compose-airflow.yml** (Version 3.7):

```
version: '3.7'
services:
  scheduler:
    image: credivo-data/airflow
    container_name: ${AIRFLOW_SCHEDULER_CONTAINER_NAME}
    hostname: ${AIRFLOW_SCHEDULER_CONTAINER_NAME}
    command: scheduler
    restart: always
    environment:
      - AIRFLOW__CORE__SQLALCHEMY_CONN=postgresql+psycopg2://airflow:airflow@postgres/airflow
      - AIRFLOW__CORE__EXECUTOR=LocalExecutor
      - POSTGRES_USER=${POSTGRES_USER}
      - POSTGRES_PASSWORD=${POSTGRES_PASSWORD}
      - POSTGRES_DB=${POSTGRES_DB}
      - POSTGRES_CONTAINER_NAME=${POSTGRES_CONTAINER_NAME}
      - POSTGRES_PORT=${POSTGRES_PORT}
      - SPARK_MASTER_HOST_NAME=${SPARK_MASTER_HOST_NAME}
      - SPARK_MASTER_PORT=${SPARK_MASTER_PORT}
    volumes:
      - ./dags:/opt/airflow/dags
      - ./logs:/opt/airflow/logs
      - ./spark-scripts:/spark-scripts
      - ./scripts:/scripts
  webserver:
    image: credivo-data/airflow
    container_name: ${AIRFLOW_WEBSERVER_CONTAINER_NAME}
    hostname: ${AIRFLOW_WEBSERVER_CONTAINER_NAME}
```

**docker-compose-postgres.yml** (Version 3.8):

```
version: '3.8'
services:
  credivo-data-postgres:
    image: postgres:11
    container_name: ${POSTGRES_CONTAINER_NAME}
    restart: unless-stopped
    hostname: ${POSTGRES_CONTAINER_NAME}
    networks:
      - credivo-network
    environment:
      - POSTGRES_PASSWORD=${POSTGRES_PASSWORD}
      - POSTGRES_USER=${POSTGRES_USER}
      - POSTGRES_DB=${POSTGRES_DB}
      - PGDATA=/var/lib/postgresql/data/pgdata
    volumes:
      - ./sql:/sql
      - ./data:/data
    ports:
      - ${POSTGRES_PORT}:5432
  networks:
    credivo-network:
      driver: bridge
      external: true
```

# Requirement

## 3. Airflow DAG (Directed Acyclic Graph):

- File:
  - [run]credivo-dag-gcs.py
- Steps:
  - Define an Airflow DAG named **move\_mysql\_to\_gbg** for loading raw data from **MySQL to Google Cloud Storage (Raw Data)** and **BigQuery (Raw Data)**
  - Import Variable and define variable in DAG file such as **BUCKET\_NAME**, **SQL\_QUERY**, **FILENAME**, **GCP\_CONN\_ID**, **MYSQL\_CONN\_ID**, and **DESTINATION\_DATASET\_TABLE**
  - Use **MySQLToGCSEoperator** to load raw data from **MySQL to Google Cloud Storage**
  - Use **GCSToBigQueryOperator** to load raw data from **Google Cloud Storage to BigQuery**
  - Schedules the DAG to run daily and set dependencies

```
[run]credivo-dag-gcs.py 4, M X
credivo_spark_airflow > dags > [run]credivo-dag-gcs.py > ...
3   import os
4   from airflow.providers.google.cloud.transfers.mysql_to_gcs import MySQLToGCSEoperator
5   from airflow.providers.google.cloud.transfers.gcs_to_bigquery import GCSToBigQueryoperator
6   from airflow.models import Variable
7
8   #dag arguments
9   default_args = {
10      'owner': 'ariq',
11      'start_date': datetime[2024, 1, 1],
12      'retries': 1,
13      'retry_delay': timedelta(minutes=5)
14  }
15
16  #define dag
17  dag = DAG('move_mysql_to_gbg',
18            schedule_interval= "@daily",
19            default_args= default_args,
20            description="Moving Data from MySQL to BigQuery",
21            dagrun_timeout=timedelta(minutes=60)
22          )
23
24  BUCKET_NAME = Variable.get("bucket_name")
25  SQL_QUERY = Variable.get("simple_query_all")
26  FILENAME = Variable.get("gcs_file")
27  GCP_CONN_ID = Variable.get("gcp_conn_id")
28  MYSQL_CONN_ID = Variable.get("mysql_conn_id")
29  DESTINATION_DATASET_TABLE = Variable.get("destination_dataset_table")
```

The screenshot shows the Airflow web interface with the following details:

- Header:** Airflow, DAGs, Cluster Activity, Datasets, Security, Browse, Admin, Docs.
- DAGs Overview:** All 1, Active 1, Paused 0, Running 16, Failed 0.
- DAG Details:** move\_mysql\_to\_gbg (Owner: ariq, Runs: 8, 16, Schedule: @daily).

# Requirement

## 4. Environment:

- File:
  - .env

## 5. Dockerfile:

- File:
  - Dockerfile.airflow-arm
  - Dockerfile.jupyter

### • Descriptions:

- Utilizes the Apache Airflow version 2.7.1
- Installs dependencies

## 6. Makefile:

- File:
  - Makefile
- Commands:
  - run-all: Starts all the containers
  - stop-all: Stops all the containers

```
.env
credivo_spark_airflow > .env
1 POSTGRES_USER=credivo
2 POSTGRES_PASSWORD=credivo
3 POSTGRES_DB=postgres_db
4 POSTGRES_PORT=5432
5 POSTGRES_CONTAINER_NAME=credivo-postgres
6
7 AIRFLOW_SCHEDULER_CONTAINER_NAME=credivo-airflow-scheduler
8 AIRFLOW_WEBSERVER_CONTAINER_NAME=credivo-airflow-webserver
9 AIRFLOW_WEBSERVER_PORT=8080
10
11 JUPYTER_CONTAINER_NAME=credivo-jupyter
12 JUPYTER_PORT=9999
13
```

```
Dockerfile.airflow-arm
credivo_spark_airflow > docker > Dockerfile.airflow-arm > ...
1 FROM apache/airflow:2.7.1-python3.9@sha256:faddcd177ae17f604f587aa8761bbd398685e00d379a2da7eab1c4ca
2 USER root
3
4 # Install OpenJDK-11
5 RUN apt update && \
6     apt-get install -y openjdk-11-jdk && \
7     apt-get install -y ant && \
8     apt-get install -y procps && \
9     apt-get clean;
10
11 # Set JAVA_HOME
12 ENV JAVA_HOME /usr/lib/jvm/java-11-openjdk-arm64
13 RUN export JAVA_HOME
14
15 USER airflow
16
17 RUN pip install \
18     lxml \
19     pyspark==3.3.2 \
20     apache-airflow-providers-apache-spark \
21     requests \
22     pandas
23
24 COPY --chown=airflow:root ./dags /opt/airflow/dags
```

```
Makefile
credivo_spark_airflow > Makefile
1 include .env
2 .PHONY: run-all
3
4 run-all: postgres airflow
5
6 .PHONY: stop-all
7
8 stop-all:
9     @docker-compose -f ./docker/docker-compose-airflow.yml stop
10    @docker-compose -f ./docker/docker-compose-postgres.yml stop
11
12 docker-build-arm:
13     @echo '_____
14     @echo 'Building Docker Images ...'
15     @echo '_____
16     @docker network inspect credivo-network >/dev/null 2>&1 || docker network create credivo-network
17     @echo '_____
18     @docker build -t credivo-data/spark -f ./docker/Dockerfile.spark .
19     @echo '_____
20     @docker build -t credivo-data/airflow -f ./docker/Dockerfile.airflow-arm .
21     @echo '_____
22     @docker build -t credivo-data/jupyter -f ./docker/Dockerfile.jupyter .
23     @echo '=====
```

# Requirement

## 7. Google Cloud Bucket

- Description:
  - Create and define bucket name as crdivo\_lake
  - Location type : Region
  - Location : asia-southeast2 (Jakarta)
  - Storage Class : Standard (We consider to use standard because of the usage activity is actively used)

## 8. BigQuery Dataset

- Description:
  - Create New dataset and define name as CREDIVO\_DW to store Raw Data and Transformed Data
  - Data location : asia-southeast2 (Jakarta)

The image shows two screenshots from the Google Cloud Platform interface.

**Bucket details (crdivo\_lake):**

Configuration	Value
Location	asia-southeast2 (Jakarta)
Storage class	Standard
Public access	Not public
Protection	None

**CREDIVO\_DW (Dataset info):**

Dataset ID	corporate-digital.CREDIVO_DW
Created	Jan 22, 2024, 4:41:44 PM UTC+7
Default table expiration	Never
Last modified	Jan 22, 2024, 4:41:44 PM UTC+7
Data location	asia-southeast2
Description	
Default collation	
Default rounding mode	ROUNDING_MODE_UNSPECIFIED
Time travel window	7 days
Storage billing model	LOGICAL
Case insensitive	false
Labels	
Tags	

# Requirement

## 9. Cloud Function

- Description:
  - Create new function named `credivo_transform` to transform data that stored in Google Cloud Storage to BigQuery
  - Region : `asia-southeast2`
  - Memory allocated : `2 GiB`
  - CPU : `1`
  - Timeout : `60 seconds`
  - Minimum instances : `0`
  - Maximum instances: `100`

## 10. Service Account for Credentials

- Description:
  - Create and define service account name as `local-airflow-mysql-gcs` to generate google cloud credential key

The screenshot shows the 'Function details' page for the 'credivo\_transform' function. The function is a 2nd gen type, deployed at Feb 3, 2024, 3:36:24 PM, with a URL: [https://asia-southeast2-corporate-digital.cloudfunctions.net/credivo\\_transform](https://asia-southeast2-corporate-digital.cloudfunctions.net/credivo_transform). The 'DETAILS' tab is selected, showing the following configuration:

General Information	Networking Settings
Last deployed: February 3, 2024 at 3:36:24 PM GMT+7	Ingress settings: Allow all traffic
Region: asia-southeast2	VPC connector: -
Memory allocated: 2 GiB	VPC connector egress: -
CPU: 1	routing
Timeout: 60 seconds	
Minimum instances: 0	
Maximum instances: 100	
Concurrency: 1	
Service account: <a href="#">compute@developer.gserviceaccount.com</a>	
Build Worker Pools: -	
Container build log: <a href="#">37af18a7-aa3f-4bb9-8648-478fc8bff7ec</a>	

The screenshot shows the 'local-airflow-mysql-gcs' service account's 'KEYS' tab. The 'KEYS' section displays a single active key:

Type	Status	Key
	Active	0b6117488a7b8b62b4be3c249e6ad5db1bd604f5

A warning message states: "Service account keys could pose a security risk if compromised. We recommend using service account credentials instead. Learn more about authenticating service accounts on Google Cloud [here](#)".

# Requirement

## 11. Data Profiling Scan on Google Cloud

Dataplex:

- Description:
  - Create and define data profile name as **DP - CRDV APPLICATION TEST**
  - Set profiling scope to **Entire Data** and sampling size to **All Data**
  - Send the profile test to BigQuery Dataset named CREDIVO\_DW and name it as DP\_{Table\_Name} (In this case, we use application\_test data to run the whole process)
  - Set repeat check on Daily

Data Profiling    [+ CREATE DATA PROFILE SCAN](#)    [+ CREATE MULTIPLE PROFILE SCANS](#)    [REFRESH](#)

Analyze the profile of your Dataplex managed data by configuring and scheduling checks over your data.

Display name	Last run	Labels	Table name	Incremental column	Schedule	⋮
DP - CRDVO APPLICATION TEST PROFILE	18 hours ago	None	<a href="#">raw_application_test</a>		Every day at 12:00 AM Etc/GMT+8	⋮

# Output

Buckets > crdovo\_lake > raw\_application\_test

**LIVE OBJECT**

**VERSION HISTORY**

**DOWNLOAD** **EDIT METADATA** **EDIT ACCESS** **DELETE**

**Overview**

Type	text/csv
Size	25.4 MB
Created	Feb 3, 2024, 5:07:38 PM
Last modified	Feb 3, 2024, 5:07:38 PM
Storage class	Standard
Custom time	—
Public URL	Not applicable
Authenticated URL	<a href="https://storage.cloud.google.com/crdovo_lake/raw_application_test">https://storage.cloud.google.com/crdovo_lake/raw_application_test</a>
gsutil URI	gs://crdovo_lake/raw_application_test

**Permissions**

Public access: Not public

**Protection**

Version history: —

Retention expiration time: None

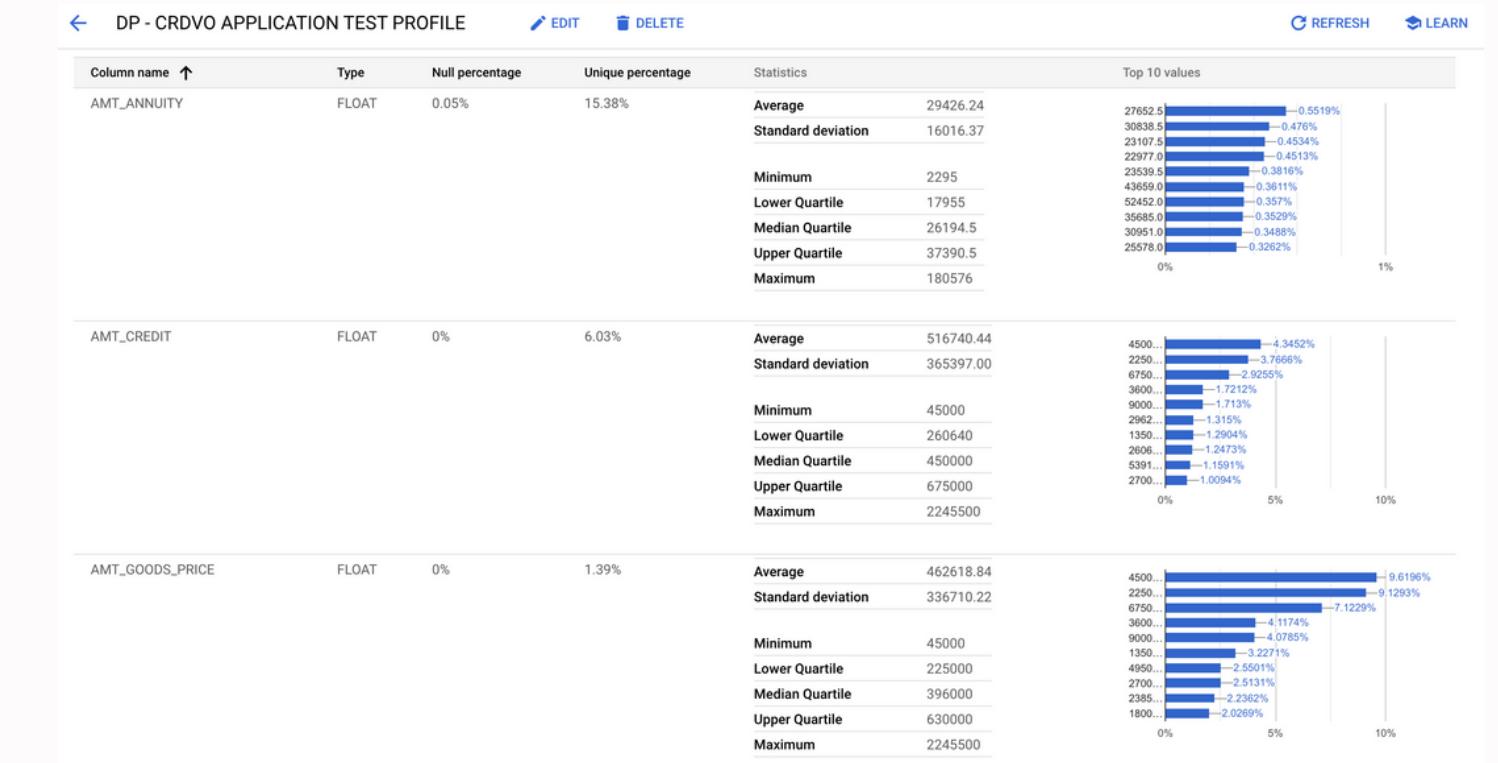
Object retention retain until time: None

Bucket retention retain until time: None

Hold status: None

Encryption type: Google-managed

Data Lake in Google Cloud Storage



Data Profiling in Dataplex

Explorer

application\_test\_clean

SCHEMA

Field name	Type	Mode	Key	Collation	Default Value	Policy Tags	Description
SK_ID_CURR	INTEGER	NULLABLE	-	-	-	-	-
NAME_CONTRACT_TYPE	STRING	NULLABLE	-	-	-	-	-
CODE_GENDER	STRING	NULLABLE	-	-	-	-	-
FLAG_OWN_CAR	STRING	NULLABLE	-	-	-	-	-
FLAG_OWN_REALTY	STRING	NULLABLE	-	-	-	-	-
CNT_CHILDREN	INTEGER	NULLABLE	-	-	-	-	-
AMT_INCOME_TOTAL	FLOAT	NULLABLE	-	-	-	-	-
AMT_CREDIT	FLOAT	NULLABLE	-	-	-	-	-
AMT_ANNUITY	FLOAT	NULLABLE	-	-	-	-	-
AMT_GOODS_PRICE	FLOAT	NULLABLE	-	-	-	-	-
NAME_TYPE_SUITE	STRING	NULLABLE	-	-	-	-	-
NAME_INCOME_TYPE	STRING	NULLABLE	-	-	-	-	-
NAME_EDUCATION_TYPE	STRING	NULLABLE	-	-	-	-	-
NAME_FAMILY_STATUS	STRING	NULLABLE	-	-	-	-	-
NAME_HOUSING_TYPE	STRING	NULLABLE	-	-	-	-	-

SUMMARY

application\_test\_clean

corporate-digital.CREDIVO\_DW

Last modified: Feb 3, 2024, 5:18:44PM UTC+7

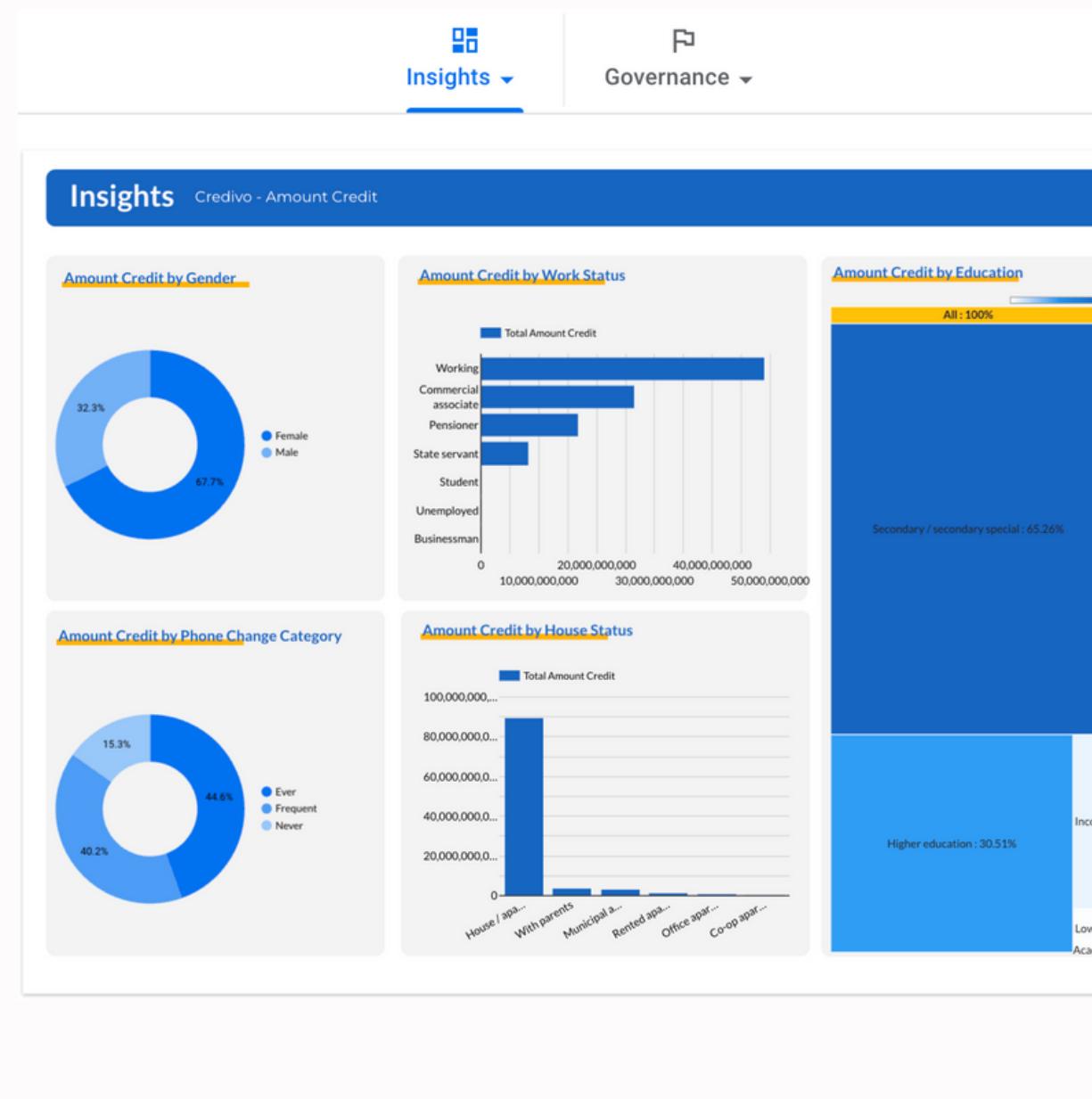
Data location: asia-southeast2

**VIEW ROW ACCESS POLICIES**

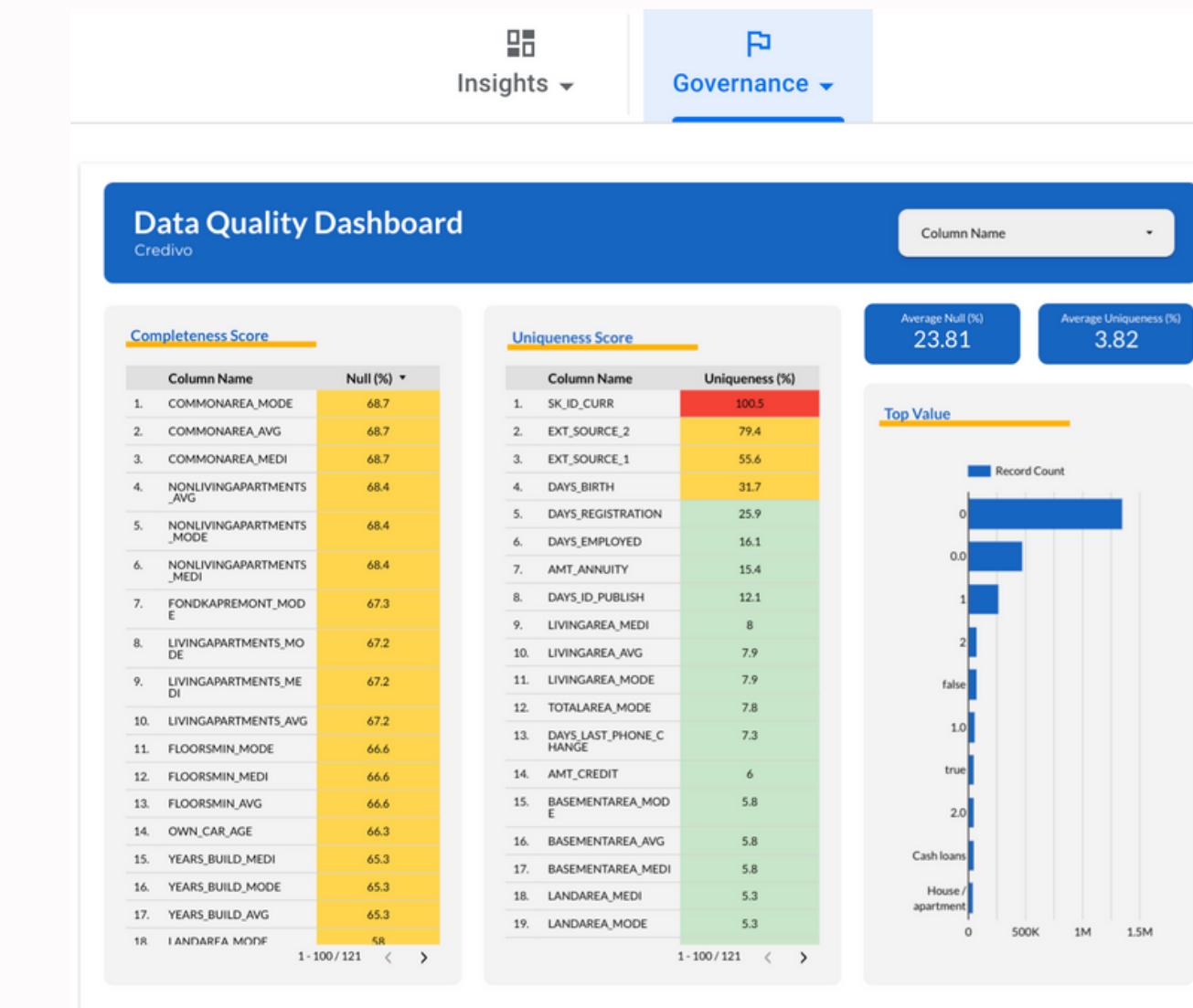
Raw and Clean Data in BigQuery

# Output

Insight Dashboard



Data Quality Dashboard



A wide-angle photograph of a modern office space. The room is filled with natural light from large windows on the left. The walls are covered in lush green plants and vines. In the center, there's a large, open-plan area with several round tables and chairs. To the right, there are more seating areas with large, comfortable couches. The ceiling is high and features exposed pipes and ductwork, along with some hanging lights. The overall atmosphere is bright, airy, and eco-friendly.

**THANK YOU**