World Bank Global Health - Data Analysis and Visualization

USING GOOGLE CLOUD PLATFORM (GCP)

Table of Contents

1 Project Goal

The project is to explore the World Bank Global Health dataset available from Google Dataset https://console.cloud.google.com/marketplace/product/the-world-bank/global-health.

This project was a part of the **Grace Hoppers TechPathways - Data Engineering Program**.

As part of the project, we were asked to create a CI/CD pipeline based on the architecture given in **Figure 1** to read data from dataset and build visualizations on it.

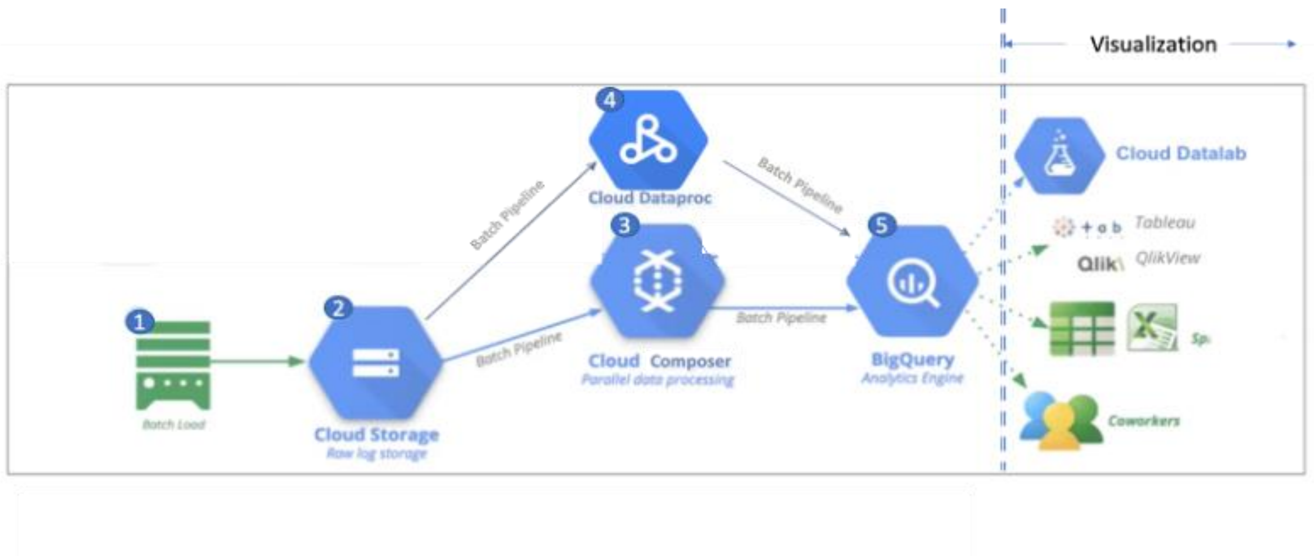

Figure 1: Architecture of the CI/CD Pipeline

The following points must be considered while developing this pipeline:

1. It must be optimized to be able to handle cost and scaling
2. The following insights had to be produced on the clean dataset using Datalab or other tools:
   a) What is the average age of males and females in different countries around the world
   b) Correlation between Health Expenditure and survival ages in the two genders around the world
   c) Correlation between School enrollment and unemployment for males and females around the world
   d) Average age of first pregnancy around the world
   e) Analysis of age of marriage and infant mortality rate across different demograpics

2 Understanding the Architecture

As per the architecture explained in the above Figure 1, the following objectives must be met:

- We have to load the WorldBank dataset in BigQuery using a CI/CD pipeline.
- The Pipeline must contain the Google Composer and Dataproc Cluster.
- Files which are not dependent on each other must be read parallelly and loaded to Bigquery.
- The data loaded in bigquery must be cleaned, transformed and easy to analyze.
- Bigquery optimization using Clustering and Partitioning must be done.

3 Implementation

The implementation was broken down into the following tasks. Every task was implemented using Google Cloud Technologies:

- Downloading Data from the Google Dataset as CSV Files to Google Storage Bucket
- Understanding the Data
- Creating Google Composer
- Creating a CI/CD Pipeline using Cloud Build
- Visualizing Data using DataLab

3.1 Downloading Data

As the very first step, data is downloaded from the Google Dataset to a Google Storage Bucket. It consists of the following csv files

1. Wh_country_series_definition.csv
2. Wh_country_summary.csv
3. Wh_health_nutrition_population.csv
4. Wh_series_summary.csv
5. Wh_series_times.csv

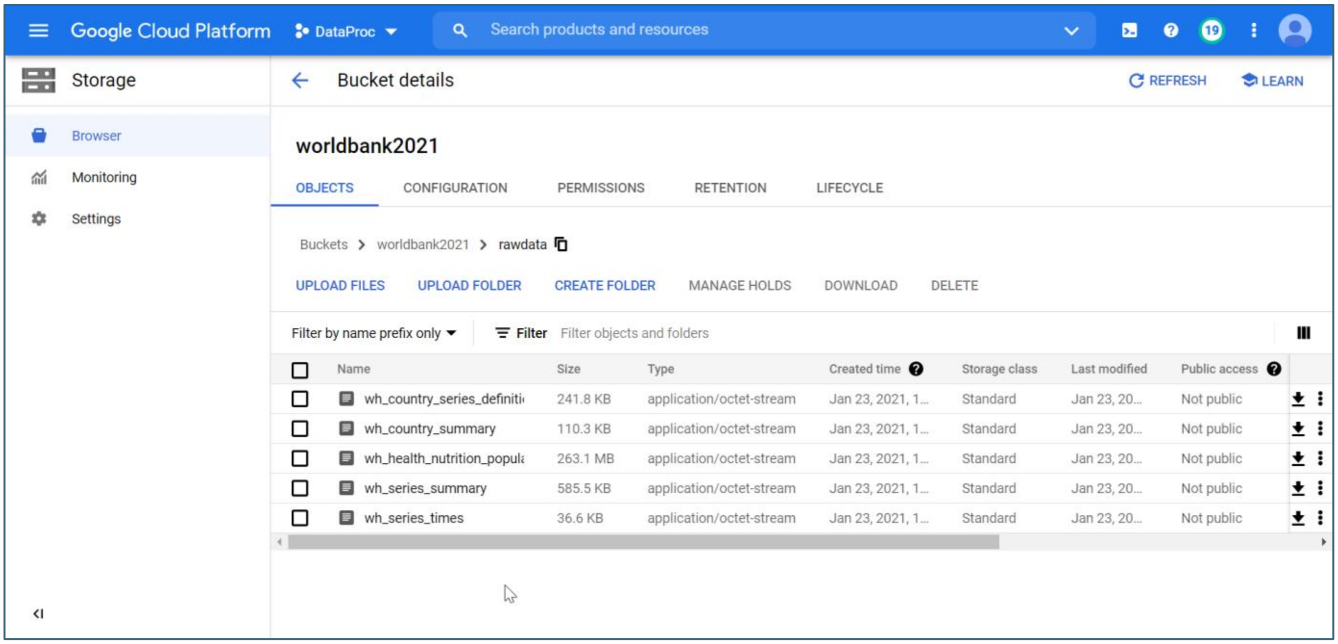

Figure 2: Google Storage Bucket

3.2 Understanding the Data

The World Bank data consists of demographic and other statistical data related to Population, Employment, Health, GDP, Energy Consumption, etc. for all the countries from the year 1960 to 2018.

From the five files downloaded in section 3.1, the *health_nutrition_population.csv* file would be used for all the data exploration tasks as that contains all the values for the various indicators. The rest of the files was used to understand the indicators for various countries.

3.3 Creating the CI/CD Pipeline

The CI/CD Pipeline is created using the Google Cloud Composer and Cloud Build. A composer is created as shown in Figure below:

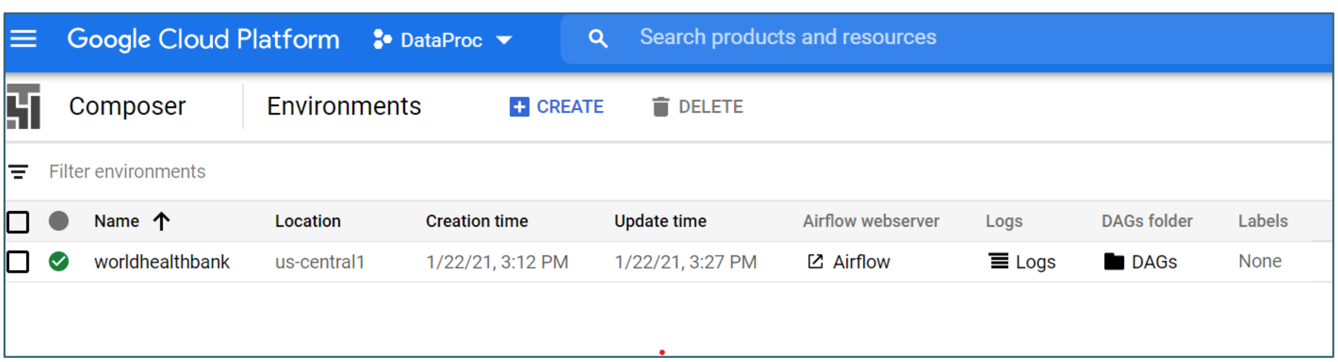

Figure 3: Google Composer

The Cloud Build trigger is created as shown below. This trigger is integrated with my Github repository https://github.com/Sadiya-Dalvi/GoogleComposer. Whenever a new dag file is checkedin to the Master branch a Cloud Build job is triggered to sync the Airflow Cloud Storage bucket. Once the dag is detected by the composer and the airflow, the pipeline gets triggered.

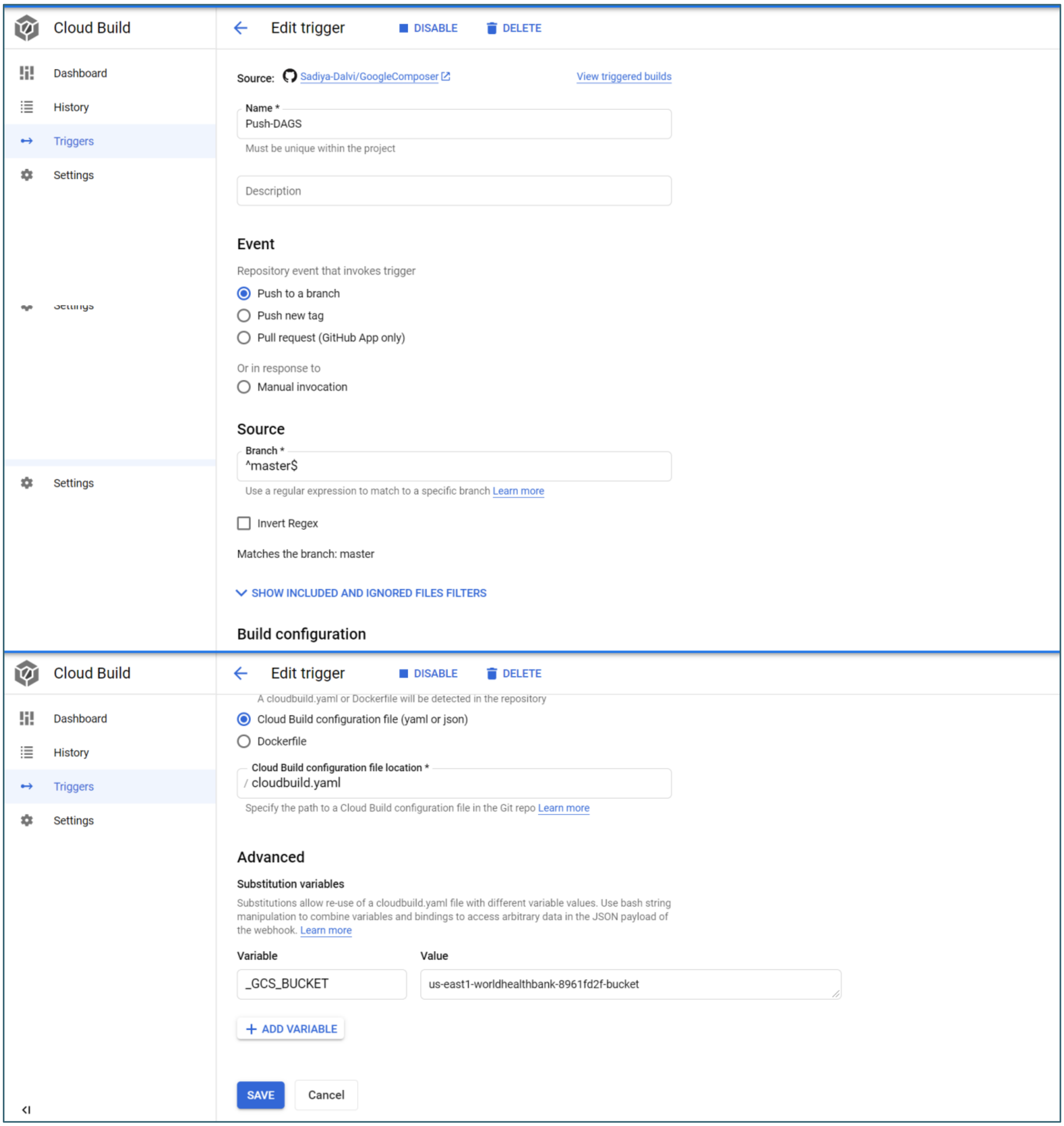

Figure 4: CloudBuild

3.3.1 Cloudbuild.yaml

```
steps:
          - name: ubuntu
            args: ['bash', '-c', "echo '$COMMIT_SHA' > REVISION.txt"]
          - name: gcr.io/cloud-builders/gsutil
            args:
              - '-m'
              - 'rsync'
              - '-d'
              - '-r'
              - 'dags'
              - 'gs://${_GCS_BUCKET}/dags'
          - name: gcr.io/cloud-builders/gsutil
            args:
              - '-m'
              - 'rsync'
              - '-d'
              - '-r'
              - 'plugins'
              - 'gs://${_GCS_BUCKET}/plugins'
```

3.3.2 Composer DAG 'load_gh_project_dag.py'

Created the following pipeline using google composer and apache airflow where the tasks are given below. The DAG code is as below:

```
from __future__ import print_function

import datetime

from airflow import models
from airflow.operators import bash_operator
from airflow.operators import python_operator
from airflow.contrib.operators import dataproc_operator
from airflow.utils import trigger_rule
from airflow.utils.dates import days_ago

yesterday = datetime.datetime.combine(
    datetime.datetime.today() - datetime.timedelta(1),
    datetime.datetime.min.time())

data_set = "worldbankhealth23012021"

default_dag_args = {
    # Setting start date as yesterday starts the DAG immediately when it is
```

```
    # detected in the Cloud Storage bucket.
    'start_date': datetime.datetime(2021, 1, 22),
}

with models.DAG(
        'Project_WH_Parallel_Datapipeline',
        schedule_interval=datetime.timedelta(days=1),
        default_args=default_dag_args) as dag:


    def greeting():
        import logging
        logging.info('Hello DataPipeline for World Health Data!')


    create_dataproc_cluster = dataproc_operator.DataprocClusterCreateOperator(
       task_id='create_dataproc_cluster',
       project_id='dataproc-300110',
       cluster_name='cluster-58-wb',
       num_workers=2,
       region='us-east1',
       init_actions_uris=['gs://worldbank2021/code/init_cluster.sh'],
       master_machine_type='n1-standard-2',
       worker_machine_type='n1-standard-2'
    )

    dataproc_pyspark_1 = dataproc_operator.DataProcPySparkOperator(
        task_id='Load_BQ_spark_job_1',
        main='gs://worldbank2021/code/dataproc_load_bq.py',
        cluster_name='cluster-58-wb',
        region='us-east1',
        arguments=['wh_country_series_definition',data_set],
        dataproc_pyspark_jars=['gs://spark-lib/bigquery/spark-bigquery-
latest.jar']
    )

    dataproc_pyspark_2 = dataproc_operator.DataProcPySparkOperator(
        task_id='Load_BQ_spark_job_2',
        main='gs://worldbank2021/code/dataproc_load_bq.py',
        cluster_name='cluster-58-wb',
        region='us-east1',
        arguments=['wh_country_summary',data_set],
        dataproc_pyspark_jars=['gs://spark-lib/bigquery/spark-bigquery-
latest.jar']
    )

    dataproc_pyspark_3 = dataproc_operator.DataProcPySparkOperator(
        task_id='Load_BQ_spark_job_3',
        # call the py file for processing
        #    main='gs://dataproc-nyc-taxi-2020/code_deploy/dataproc_wb.py',
        main='gs://worldbank2021/code/dataproc_load_bq.py',
        cluster_name='cluster-58-wb',
        region='us-east1',
        arguments=['wh_health_nutrition_population',data_set],
        dataproc_pyspark_jars=['gs://spark-lib/bigquery/spark-bigquery-
latest.jar']
```

```
    )

    dataproc_pyspark_4 = dataproc_operator.DataProcPySparkOperator(
        task_id='Load_BQ_spark_job_4',
        # call the py file for processing
        #    main='gs://dataproc-nyc-taxi-2020/code_deploy/dataproc_wb.py',
        main='gs://worldbank2021/code/dataproc_load_bq.py',
        cluster_name='cluster-58-wb',
        region='us-east1',
        arguments=['wh_series_summary',data_set],
        dataproc_pyspark_jars=['gs://spark-lib/bigquery/spark-bigquery-
latest.jar']
    )

    dataproc_pyspark_5 = dataproc_operator.DataProcPySparkOperator(
        task_id='Load_BQ_spark_job_5',
        # call the py file for processing
        #    main='gs://dataproc-nyc-taxi-2020/code_deploy/dataproc_wb.py',
        main='gs://worldbank2021/code/dataproc_load_bq.py',
        cluster_name='cluster-58-wb',
        region='us-east1',
        arguments=['wh_series_times',data_set],
        dataproc_pyspark_jars=['gs://spark-lib/bigquery/spark-bigquery-
latest.jar']
    )

    delete_dataproc_cluster = dataproc_operator.DataprocClusterDeleteOperator(
        project_id='dataproc-300110',
        task_id = 'delete_dataproc_cluster',
        cluster_name='cluster-58-wb',
        region='us-east1',
        trigger_rule = trigger_rule.TriggerRule.ALL_DONE
    )

    create_dataproc_cluster >> dataproc_pyspark_1 >> delete_dataproc_cluster
    create_dataproc_cluster >> dataproc_pyspark_2 >> delete_dataproc_cluster
    create_dataproc_cluster >> dataproc_pyspark_3 >> delete_dataproc_cluster
    create_dataproc_cluster >> dataproc_pyspark_4 >> delete_dataproc_cluster
    create_dataproc_cluster >> dataproc_pyspark_5 >> delete_dataproc_cluster
```

3.3.3 Dataproc_load_bq.py

```
#test
#import pyspark
import sys
from pyspark.sql import SparkSession
from pyspark.sql.types import TimestampType, StringType
import pyspark.sql.functions as F
#from google.cloud import bigquery

#import datetime
#gs://worldbank2021/rawdata/wh_country_series_definition
#gs://worldbank2021/rawdata/wh_country_summary
#gs://worldbank2021/rawdata/wh_health_nutrition_population
```

```
#gs://worldbank2021/rawdata/wh_series_summary
#gs://worldbank2021/rawdata/wh_series_times


spark = SparkSession.builder.appName('World Health Data').getOrCreate()

#nyctaxi_df = spark.read.csv("gs://dataproc-nyc-taxi-2020", header=True,
inferSchema=True)
table_name = sys.argv[1]
data_set   = sys.argv[2]
file_path = "gs://worldbank2021/rawdata/" + table_name
print("File name is " + file_path)
#"gs://worldbank2021/rawdata/health_nutrition_population"

df = spark.read.csv(file_path, header=True, inferSchema=True)


df.printSchema()
#df.count()
#df.describe().show()

df = df.dropDuplicates()

#df.show()

print("There are {} rows in the Dataframe after dropping
duplicates.".format(df.count()))

bucket = "worldbank2021"
spark.conf.set('temporaryGcsBucket', bucket)

if table_name == 'wh_country_series_definitions':
    df = df.dropna(subset=['series_code'])
    print("There are {} rows in the Dataframe after dropping
nulls.".format(df.count()))

if table_name == 'wh_country_summary':
    df = df.dropna(subset=['country_code'])
    print("There are {} rows in the Dataframe after dropping
nulls.".format(df.count()))

if table_name == 'wh_series_summary':
    df = df.dropna(subset=['series_code'])
    print("There are {} rows in the Dataframe after dropping
nulls.".format(df.count()))

if table_name == 'wh_series_times':
    df = df.dropna(subset=['series_code'])
    print("There are {} rows in the Dataframe after dropping
nulls.".format(df.count()))

if table_name == 'wh_health_nutrition_population':
    df = df.dropna(subset=['indicator_code'])
    print("There are {} rows in the Dataframe after dropping
nulls.".format(df.count()))
```

```
#df.show()



#df.createOrReplaceTempView('wh_table')

# Perform filter by year on country = ARB.
#filter_rows = spark.sql(
 #       "SELECT country_name, indicator_name, value, year FROM wh_table where
country_code = 'ARB'")
#filter_rows.show()
#filter_rows.printSchema()

if table_name == 'wh_health_nutrition_population':
    df = df.withColumn('dateYear',
                       F.to_date(F.col('year').cast(StringType()),'yyyy'))

#filter_rows_date.show()
#filter_rows_date.printSchema()

#df.show()

# Construct a BigQuery client object.
#bq_client = bigquery.Client()

#table_id = 'dataproc-300110:worldbankhealth' + table_name

# If the table does not exist, delete_table raises
# google.api_core.exceptions.NotFound unless not_found_ok is True.
#bq_client.delete_table(table_id, not_found_ok=True)  # Make an API request.
#print("Deleted table '{}'.".format(table_id))

bq_table_name = data_set + '.' + table_name

if table_name == 'wh_health_nutrition_population':
    df.write.format('bigquery') \
        .option('table', bq_table_name) \
        .option('partitionField','dateYear') \
        .option('clusteredFields','indicator_code')\
        .save()
else:
    df.write.format('bigquery') \
        .option('table', bq_table_name) \
        .save()
```

3.3.4 Task Descriptions

1. **Task id(create_dataproc_cluster)** - Created a dataproc cluster

   Logs from Apache Airflow are given below:

```
--------------------------------------------------------------------------------
[2021-01-23 07:04:42,699] {taskinstance.py:882} INFO - Starting attempt 1 of 1
[2021-01-23 07:04:42,700] {taskinstance.py:883} INFO -
--------------------------------------------------------------------------------
[2021-01-23 07:04:42,754] {taskinstance.py:902} INFO - Executing
<Task(DataprocClusterCreateOperator): create_dataproc_cluster> on 2021-01-
22T00:00:00+00:00
[2021-01-23 07:04:42,760] {standard_task_runner.py:54} INFO - Started process 9759 to
run task
[2021-01-23 07:04:42,868] {standard_task_runner.py:77} INFO - Running: ['airflow',
'run', 'Project_WH_Parallel_Datapipeline', 'create_dataproc_cluster', '2021-01-
22T00:00:00+00:00', '--job_id', '285', '--pool', 'default_pool', '--raw', '-sd',
'DAGS_FOLDER/load_gh_project_dag.py', '--cfg_path', '/tmp/tmp90u77lsg']
```

```
[2021-01-23 07:04:42,870] {standard_task_runner.py:78} INFO - Job 285: Subtask
create_dataproc_cluster
[2021-01-23 07:04:43,618] {logging_mixin.py:112} INFO - Running <TaskInstance:
Project_WH_Parallel_Datapipeline.create_dataproc_cluster 2021-01-22T00:00:00+00:00
[running]> on host airflow-worker-84bdd7d564-7m64g
[2021-01-23 07:04:43,967] {dataproc_operator.py:447} INFO - Creating cluster:
cluster-58-wb@-@{"workflow": "Project_WH_Parallel_Datapipeline", "task-id":
"create_dataproc_cluster", "execution-date": "2021-01-22T00:00:00+00:00"}
[2021-01-23 07:04:43,977] {gcp_api_base_hook.py:145} INFO - Getting connection using
`google.auth.default()` since no key file is defined for hook.@-@{"workflow":
"Project_WH_Parallel_Datapipeline", "task-id": "create_dataproc_cluster", "execution-
date": "2021-01-22T00:00:00+00:00"}
[2021-01-23 07:04:45,915] {gcp_api_base_hook.py:145} INFO - Getting connection using
`google.auth.default()` since no key file is defined for hook.@-@{"workflow":
"Project_WH_Parallel_Datapipeline", "task-id": "create_dataproc_cluster", "execution-
date": "2021-01-22T00:00:00+00:00"}
[2021-01-23 07:04:46,151] {gcp_dataproc_hook.py:250} INFO - Waiting for Dataproc
Operation projects/dataproc-300110/regions/us-east1/operations/7e9f1383-54db-31c6-
9ab6-6fedec239487 to finish@-@{"workflow": "Project_WH_Parallel_Datapipeline", "task-
id": "create_dataproc_cluster", "execution-date": "2021-01-22T00:00:00+00:00"}
[2021-01-23 07:06:56,766] {gcp_dataproc_hook.py:275} INFO - Dataproc Operation
projects/dataproc-300110/regions/us-east1/operations/7e9f1383-54db-31c6-9ab6-
6fedec239487 done@-@{"workflow": "Project_WH_Parallel_Datapipeline", "task-id":
"create_dataproc_cluster", "execution-date": "2021-01-22T00:00:00+00:00"}
[2021-01-23 07:06:56,907] {taskinstance.py:1071} INFO - Marking task as
SUCCESS.dag_id=Project_WH_Parallel_Datapipeline, task_id=create_dataproc_cluster,
execution_date=20210122T000000, start_date=20210123T070442, end_date=20210123T070656
[2021-01-23 07:06:58,074] {local_task_job.py:102} INFO - Task exited with return code
0
```

2. **Task id(dataproc_pyspark_1)** – Ran 'Load_BQ_spark_job_1' job. This job executes the dataproc_load_bq.py script to read data from **wh_country_series_definition.csv** file, extract and cleaning it to drop duplicates, dropping null values and writes it to bigquery table.

   Log Output from the Dataproc Cluster job is given below:

```
21/01/23 07:17:17 WARN org.apache.spark.scheduler.cluster.YarnScheduler: Initial job
has not accepted any resources; check your cluster UI to ensure that workers are
registered and have sufficient resources
root
 |-- country_code: string (nullable = true)
 |-- series_code: string (nullable = true)
 |-- description: string (nullable = true)

There are 3368 rows in the Dataframe after dropping duplicates.
21/01/23 07:17:38 INFO com.google.cloud.spark.bigquery.BigQueryUtilScala: BigQuery
client project id is [dataproc-300110], derived from the parentProject option
21/01/23 07:22:12 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper:
Submitted load to GenericData{classInfo=[datasetId, projectId, tableId],
{datasetId=worldbankhealth23012021, projectId=dataproc-300110,
tableId=wh_country_series_definition}}. jobId: JobId{project=dataproc-300110,
job=b0e23981-04ab-4242-9429-d6dc58cee6d0, location=US}
21/01/23 07:22:19 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper: Done
loading to dataproc-300110.worldbankhealth23012021.wh_country_series_definition.
jobId: JobId{project=dataproc-300110, job=b0e23981-04ab-4242-9429-d6dc58cee6d0,
location=US}
21/01/23 07:22:20 INFO org.spark_project.jetty.server.AbstractConnector: Stopped
Spark@70e6cecb{HTTP/1.1,[http/1.1]}{0.0.0.0:4043}
Job output is complete
```

3. **Task id(dataproc_pyspark_2)** – Ran 'Load_BQ_spark_job_2' job. This job executes the dataproc_load_bq.py script to read data from **wh_country_summary.csv** file, extract and cleaning it to drop duplicates, dropping null values and writes it to bigquery table.

Log Output from the Dataproc Cluster job is given below:

```
21/01/23 07:13:03 WARN org.apache.spark.scheduler.cluster.YarnScheduler: Initial job
has not accepted any resources; check your cluster UI to ensure that workers are
registered and have sufficient resources
root
 |-- country_code: string (nullable = true)
 |-- short_name: string (nullable = true)
 |-- table_name: string (nullable = true)
 |-- long_name: string (nullable = true)
 |-- two_alpha_code: string (nullable = true)
 |-- currency_unit: string (nullable = true)
 |-- special_notes: string (nullable = true)
 |-- region: string (nullable = true)
 |-- income_group: string (nullable = true)
 |-- wb_2_code: string (nullable = true)
 |-- national_accounts_base_year: string (nullable = true)
 |-- national_accounts_reference_year: integer (nullable = true)
 |-- sna_price_valuation: string (nullable = true)
 |-- lending_category: string (nullable = true)
 |-- other_groups: string (nullable = true)
 |-- system_of_national_accounts: string (nullable = true)
 |-- alternative_conversion_factor: string (nullable = true)
 |-- ppp_survey_year: string (nullable = true)
 |-- balance_of_payments_manual_in_use: string (nullable = true)
 |-- external_debt_reporting_status: string (nullable = true)
 |-- system_of_trade: string (nullable = true)
 |-- government_accounting_concept: string (nullable = true)
 |-- imf_data_dissemination_standard: string (nullable = true)
 |-- latest_population_census: string (nullable = true)
 |-- latest_household_survey: string (nullable = true)
 |-- source_of_most_recent_income_and_expenditure_data: string (nullable = true)
 |-- vital_registration_complete: string (nullable = true)
 |-- latest_agricultural_census: string (nullable = true)
 |-- latest_industrial_data: integer (nullable = true)
 |-- latest_trade_data: integer (nullable = true)
 |-- latest_water_withdrawal_data: string (nullable = true)

21/01/23 07:13:21 WARN org.apache.spark.util.Utils: Truncated the string
representation of a plan since it was too large. This behavior can be adjusted by
setting 'spark.debug.maxToStringFields' in SparkEnv.conf.
There are 262 rows in the Dataframe after dropping duplicates.
There are 262 rows in the Dataframe after dropping nulls.
21/01/23 07:13:33 INFO com.google.cloud.spark.bigquery.BigQueryUtilScala: BigQuery
client project id is [dataproc-300110], derived from the parentProject option
21/01/23 07:17:09 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper:
Submitted load to GenericData{classInfo=[datasetId, projectId, tableId],
{datasetId=worldbankhealth23012021, projectId=dataproc-300110,
tableId=wh_country_summary}}. jobId: JobId{project=dataproc-300110, job=609d36f4-
f90a-49e9-ab2e-9833c0cfb0e9, location=US}
21/01/23 07:17:20 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper: Done
loading to dataproc-300110.worldbankhealth23012021.wh_country_summary. jobId:
JobId{project=dataproc-300110, job=609d36f4-f90a-49e9-ab2e-9833c0cfb0e9,
location=US}
21/01/23 07:17:20 INFO org.spark_project.jetty.server.AbstractConnector: Stopped
Spark@6cf95103{HTTP/1.1,[http/1.1]}{0.0.0.0:4041}
Job output is complete
```

4. **Task id(dataproc_pyspark_3)** – Ran 'Load_BQ_spark_job_3' job. This job executes the dataproc_load_bq.py script to read data from **wh_health_nutrition_population.csv** file, extract and cleaning it to drop duplicates, dropping null values and writes it to bigquery table.

Additionally since this is the table which we would use for our visualizations, an additional column, dateYear is added to this. This is done to partition the bigquery table on this column since there was no date column available in the original table. Partitioning helps with query optimizations. Further we also use the cluster property on indicator_code column for performance improvement.

Log Output from the Dataproc Cluster job is given below:

```
21/01/23 07:08:06 INFO org.spark_project.jetty.util.log: Logging initialized
@18143ms
21/01/23 07:08:07 INFO org.spark_project.jetty.server.Server: jetty-9.3.z-SNAPSHOT,
build timestamp: unknown, git hash: unknown
21/01/23 07:08:07 INFO org.spark_project.jetty.server.Server: Started @18821ms
21/01/23 07:08:07 WARN org.apache.spark.util.Utils: Service 'SparkUI' could not
bind on port 4040. Attempting port 4041.
21/01/23 07:08:07 WARN org.apache.spark.util.Utils: Service 'SparkUI' could not
bind on port 4041. Attempting port 4042.
21/01/23 07:08:07 WARN org.apache.spark.util.Utils: Service 'SparkUI' could not
bind on port 4042. Attempting port 4043.
21/01/23 07:08:07 WARN org.apache.spark.util.Utils: Service 'SparkUI' could not
bind on port 4043. Attempting port 4044.
21/01/23 07:08:07 INFO org.spark_project.jetty.server.AbstractConnector: Started
ServerConnector@2449bef3{HTTP/1.1,[http/1.1]}{0.0.0.0:4044}
21/01/23 07:08:08 WARN org.apache.spark.scheduler.FairSchedulableBuilder: Fair
Scheduler configuration file not found so jobs will be scheduled in FIFO order. To
use fair scheduling, configure pools in fairscheduler.xml or set
spark.scheduler.allocation.file to a file that contains the configuration.
21/01/23 07:08:12 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to
ResourceManager at cluster-58-wb-m/10.142.15.199:8032
21/01/23 07:08:13 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to
Application History server at cluster-58-wb-m/10.142.15.199:10200
21/01/23 07:08:22 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl:
Submitted application application_1611385532354_0005
File name is gs://worldbank2021/rawdata/wh_health_nutrition_population
21/01/23 07:17:51 WARN org.apache.spark.scheduler.cluster.YarnScheduler: Initial
job has not accepted any resources; check your cluster UI to ensure that workers
are registered and have sufficient resources
root
 |-- country_name: string (nullable = true)
 |-- country_code: string (nullable = true)
 |-- indicator_name: string (nullable = true)
 |-- indicator_code: string (nullable = true)
 |-- value: double (nullable = true)
 |-- year: integer (nullable = true)

There are 3019732 rows in the Dataframe after dropping duplicates.
There are 3019732 rows in the Dataframe after dropping nulls.
21/01/23 07:18:56 INFO com.google.cloud.spark.bigquery.BigQueryUtilScala: BigQuery
client project id is [dataproc-300110], derived from the parentProject option
21/01/23 07:23:09 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper:
Submitted load to GenericData{classInfo=[datasetId, projectId, tableId],
{datasetId=worldbankhealth23012021, projectId=dataproc-300110,
tableId=wh_health_nutrition_population}}. jobId: JobId{project=dataproc-300110,
job=25a700b8-94c5-4052-b9af-a0a3e5da9629, location=US}
21/01/23 07:23:18 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper: Done
loading to dataproc-300110.worldbankhealth23012021.wh_health_nutrition_population.
jobId: JobId{project=dataproc-300110, job=25a700b8-94c5-4052-b9af-a0a3e5da9629,
location=US}
```

```
21/01/23 07:23:19 INFO org.spark_project.jetty.server.AbstractConnector: Stopped
Spark@2449bef3{HTTP/1.1,[http/1.1]}{0.0.0.0:4044}
Job output is complete
```

5. **Task id(dataproc_pyspark_4)** – Ran 'Load_BQ_spark_job_4' job. This job executes the dataproc_load_bq.py script to read data from **wh_series_summary.csv** file, extract and cleaning it to drop duplicates, dropping null values and writes it to bigquery table.

```
21/01/23 07:07:30 INFO org.spark_project.jetty.util.log: Logging initialized @3567ms
21/01/23 07:07:30 INFO org.spark_project.jetty.server.Server: jetty-9.3.z-SNAPSHOT,
build timestamp: unknown, git hash: unknown
21/01/23 07:07:30 INFO org.spark_project.jetty.server.Server: Started @3669ms
21/01/23 07:07:30 INFO org.spark_project.jetty.server.AbstractConnector: Started
ServerConnector@3f07e249{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
21/01/23 07:07:30 WARN org.apache.spark.scheduler.FairSchedulableBuilder: Fair
Scheduler configuration file not found so jobs will be scheduled in FIFO order. To
use fair scheduling, configure pools in fairscheduler.xml or set
spark.scheduler.allocation.file to a file that contains the configuration.
21/01/23 07:07:31 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to
ResourceManager at cluster-58-wb-m/10.142.15.199:8032
21/01/23 07:07:32 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to
Application History server at cluster-58-wb-m/10.142.15.199:10200
21/01/23 07:07:37 INFO org.apache.hadoop.yarn.client.api.impl.YarnClientImpl:
Submitted application application_1611385532354_0001
File name is gs://worldbank2021/rawdata/wh_series_summary
root
 |-- series_code: string (nullable = true)
 |-- topic: string (nullable = true)
 |-- indicator_name: string (nullable = true)
 |-- short_definition: string (nullable = true)
 |-- long_definition: string (nullable = true)
 |-- unit_of_measure: string (nullable = true)
 |-- periodicity: string (nullable = true)
 |-- base_period: string (nullable = true)
 |-- other_notes: string (nullable = true)
 |-- aggregation_method: string (nullable = true)
 |-- limitations_and_exceptions: string (nullable = true)
 |-- notes_from_original_source: string (nullable = true)
 |-- general_comments: string (nullable = true)
 |-- source: string (nullable = true)
 |-- statistical_concept_and_methodology: string (nullable = true)
 |-- development_relevance: string (nullable = true)
 |-- related_source_links: string (nullable = true)
 |-- other_web_links: string (nullable = true)
 |-- related_indicators: string (nullable = true)
 |-- license_type: string (nullable = true)

There are 542 rows in the Dataframe after dropping duplicates.
There are 542 rows in the Dataframe after dropping nulls.
21/01/23 07:08:35 INFO com.google.cloud.spark.bigquery.BigQueryUtilScala: BigQuery
client project id is [dataproc-300110], derived from the parentProject option
21/01/23 07:12:55 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper:
Submitted load to GenericData{classInfo=[datasetId, projectId, tableId],
{datasetId=worldbankhealth23012021, projectId=dataproc-300110,
tableId=wh_series_summary}}. jobId: JobId{project=dataproc-300110, job=0056f5f3-
555e-410c-8949-0c86271d9df4, location=US}
21/01/23 07:13:11 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper: Done
loading to dataproc-300110.worldbankhealth23012021.wh_series_summary. jobId:
JobId{project=dataproc-300110, job=0056f5f3-555e-410c-8949-0c86271d9df4,
location=US}
21/01/23 07:13:11 INFO org.spark_project.jetty.server.AbstractConnector: Stopped
Spark@3f07e249{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
Job output is complete
```

6. **Task id(dataproc_pyspark_5)** – Ran 'Load_BQ_spark_job_5' job. This job executes the dataproc_load_bq.py script to read data from **wh_series_times.csv** file, extract and cleaning it to drop duplicates, dropping null values and writes it to bigquery table.

```
21/01/23 07:13:16 WARN org.apache.spark.scheduler.cluster.YarnScheduler: Initial job
has not accepted any resources; check your cluster UI to ensure that workers are
registered and have sufficient resources
root
 |-- series_code: string (nullable = true)
 |-- year: integer (nullable = true)
 |-- description: string (nullable = true)

There are 420 rows in the Dataframe after dropping duplicates.
There are 420 rows in the Dataframe after dropping nulls.
21/01/23 07:13:37 INFO com.google.cloud.spark.bigquery.BigQueryUtilScala: BigQuery
client project id is [dataproc-300110], derived from the parentProject option
21/01/23 07:17:33 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper:
Submitted load to GenericData{classInfo=[datasetId, projectId, tableId],
{datasetId=worldbankhealth23012021, projectId=dataproc-300110,
tableId=wh_series_times}}. jobId: JobId{project=dataproc-300110, job=1d4dd7c9-3694-
4b05-b3f2-590b5c2aad0c, location=US}
21/01/23 07:17:46 INFO com.google.cloud.spark.bigquery.BigQueryWriteHelper: Done
loading to dataproc-300110.worldbankhealth23012021.wh_series_times. jobId:
JobId{project=dataproc-300110, job=1d4dd7c9-3694-4b05-b3f2-590b5c2aad0c,
location=US}
21/01/23 07:17:46 INFO org.spark_project.jetty.server.AbstractConnector: Stopped
Spark@77f31878{HTTP/1.1,[http/1.1]}{0.0.0.0:4042}
Job output is complete
```

7. **Task id (delete_dataproc_cluster)** - Deleted the dataproc cluster 'delete_dataproc_cluster' after the tables were loaded in Bigquery

```
[2021-01-23 07:23:32,624] {taskinstance.py:882} INFO - Starting attempt 1 of 1

[2021-01-23 07:23:32,624] {taskinstance.py:883} INFO -

--------------------------------------------------------------------------------

[2021-01-23 07:23:32,695] {taskinstance.py:902} INFO - Executing <Task(DataprocClusterDe
leteOperator): delete_dataproc_cluster> on 2021-01-22T00:00:00+00:00

[2021-01-23 07:23:32,728] {standard_task_runner.py:54} INFO - Started process 9753 to ru
n task

[2021-01-23 07:23:33,283] {standard_task_runner.py:77} INFO - Running: ['airflow', 'run'
, 'Project_WH_Parallel_Datapipeline', 'delete_dataproc_cluster', '2021-01-22T00:00:00+00
:00', '--job_id', '295', '--pool', 'default_pool', '--raw', '-sd', 'DAGS_FOLDER/load_gh_
project_dag.py', '--cfg_path', '/tmp/tmpyl03n3wb']

[2021-01-23 07:23:33,286] {standard_task_runner.py:78} INFO - Job 295: Subtask delete_da
taproc_cluster

[2021-01-23 07:23:34,232] {logging_mixin.py:112} INFO - Running <TaskInstance: Project_W
H_Parallel_Datapipeline.delete_dataproc_cluster 2021-01-22T00:00:00+00:00 [running]> on
host airflow-worker-84bdd7d564-wjwcl

[2021-01-23 07:23:34,403] {dataproc_operator.py:620} INFO - Deleting cluster: cluster-58
-wb in us-east1@-@{"workflow": "Project_WH_Parallel_Datapipeline", "task-id": "delete_da
taproc_cluster", "execution-date": "2021-01-22T00:00:00+00:00"}
```

```
[2021-01-23 07:23:34,404] {gcp_api_base_hook.py:145} INFO - Getting connection using `go
ogle.auth.default()` since no key file is defined for hook.@-@{"workflow": "Project_WH_P
arallel_Datapipeline", "task-id": "delete_dataproc_cluster", "execution-date": "2021-01-
22T00:00:00+00:00"}

[2021-01-23 07:23:34,988] {gcp_api_base_hook.py:145} INFO - Getting connection using `go
ogle.auth.default()` since no key file is defined for hook.@-@{"workflow": "Project_WH_P
arallel_Datapipeline", "task-id": "delete_dataproc_cluster", "execution-date": "2021-01-
22T00:00:00+00:00"}

[2021-01-23 07:23:35,359] {gcp_dataproc_hook.py:250} INFO - Waiting for Dataproc Operati
on projects/dataproc-300110/regions/us-east1/operations/1bd8742b-14a7-34dd-b139-675c21a3
5c74 to finish@-@{"workflow": "Project_WH_Parallel_Datapipeline", "task-id": "delete_dat
aproc_cluster", "execution-date": "2021-01-22T00:00:00+00:00"}

[2021-01-23 07:24:05,630] {gcp_dataproc_hook.py:275} INFO - Dataproc Operation projects/
dataproc-300110/regions/us-east1/operations/1bd8742b-14a7-34dd-b139-675c21a35c74 done@-@
{"workflow": "Project_WH_Parallel_Datapipeline", "task-id": "delete_dataproc_cluster", "
execution-date": "2021-01-22T00:00:00+00:00"}

[2021-01-23 07:24:05,784] {taskinstance.py:1071} INFO - Marking task as SUCCESS.dag_id=P
roject_WH_Parallel_Datapipeline, task_id=delete_dataproc_cluster, execution_date=2021012
2T000000, start_date=20210123T072332, end_date=20210123T072405

[2021-01-23 07:24:07,491] {local_task_job.py:102} INFO - Task exited with return code 0
```

3.4 Viewing the DAG Graph View in Apache Airflow

The following Figure shows the graph view of the workflow.

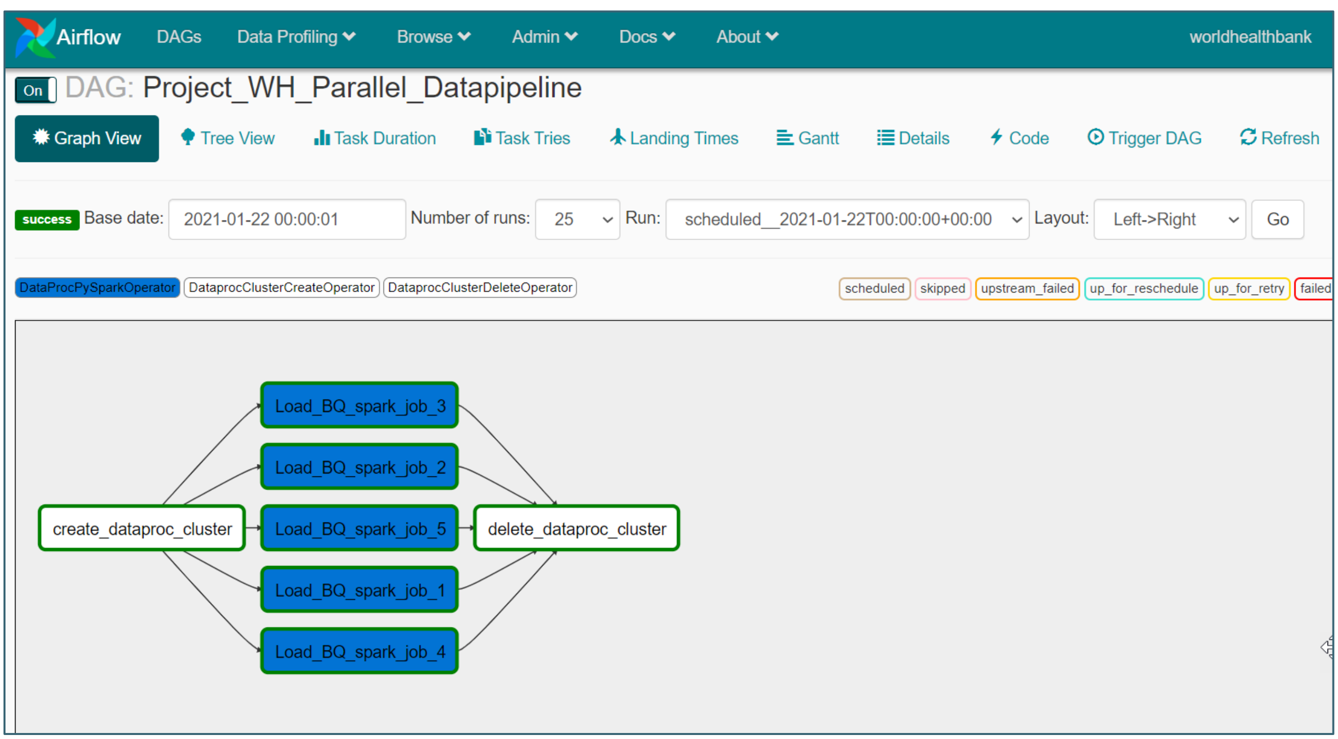

Figure 5:Graph View

3.5 Viewing the DAG Gantt View in Apache Airflow

The following figure shows the Gantt view of the workflow. Note the parallel processing of jobs. Had these jobs been run sequentially more time would have taken for running the workflow.

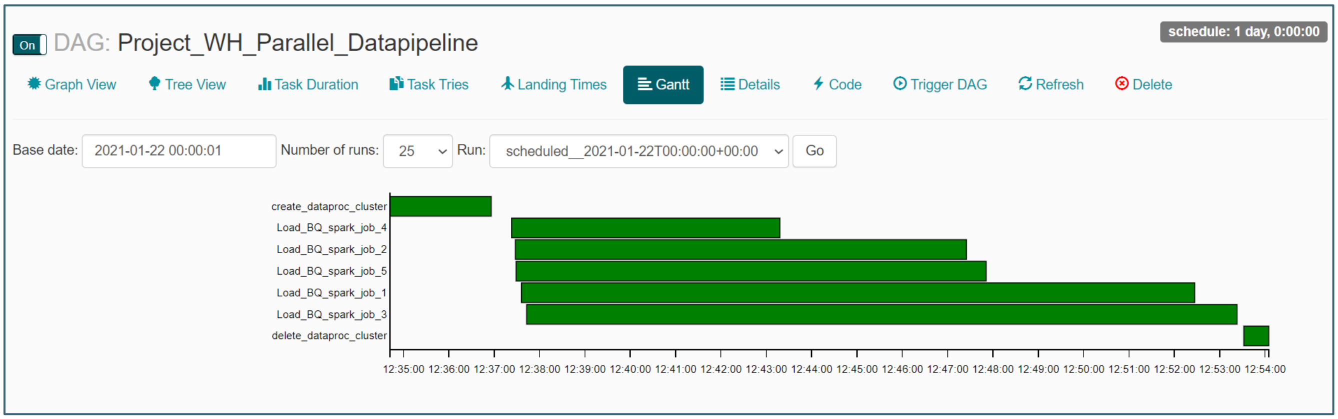

Figure 6: Gantt View

3.6 Tables loaded in Bigquery

The following Figure shows the tables loaded in bigquery. We are particularly interested in the wh_health_nutrition_population table. This is a partitioned and clustered table.

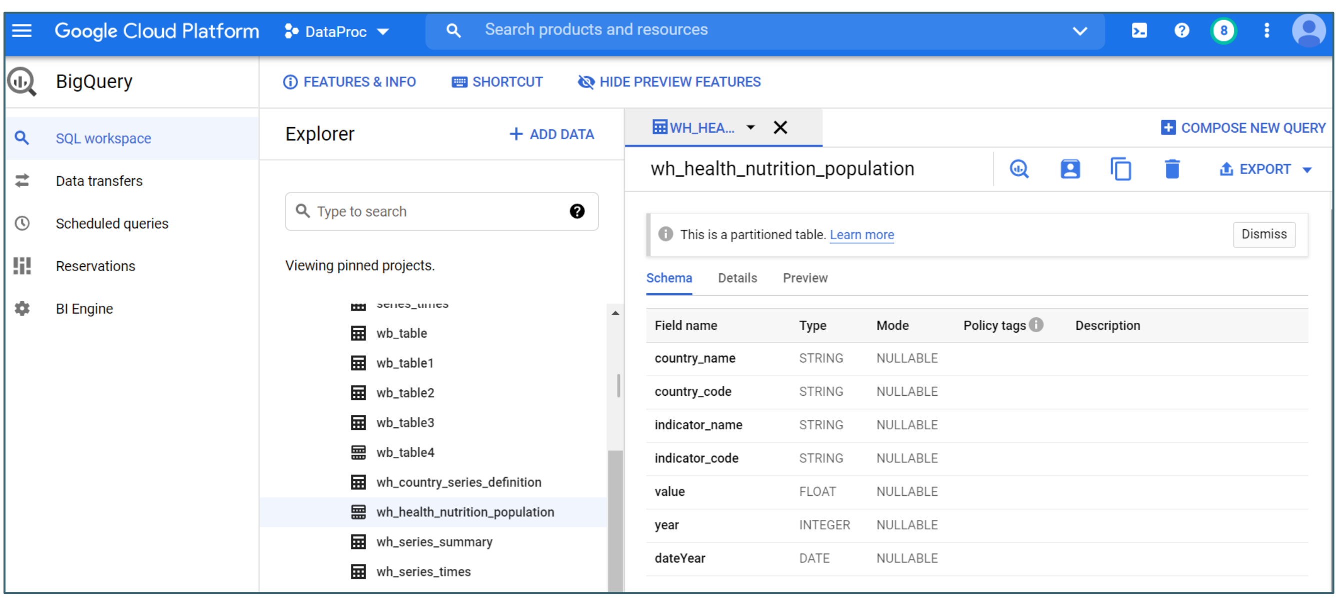

Figure 7: Tables loaded in Bigquery

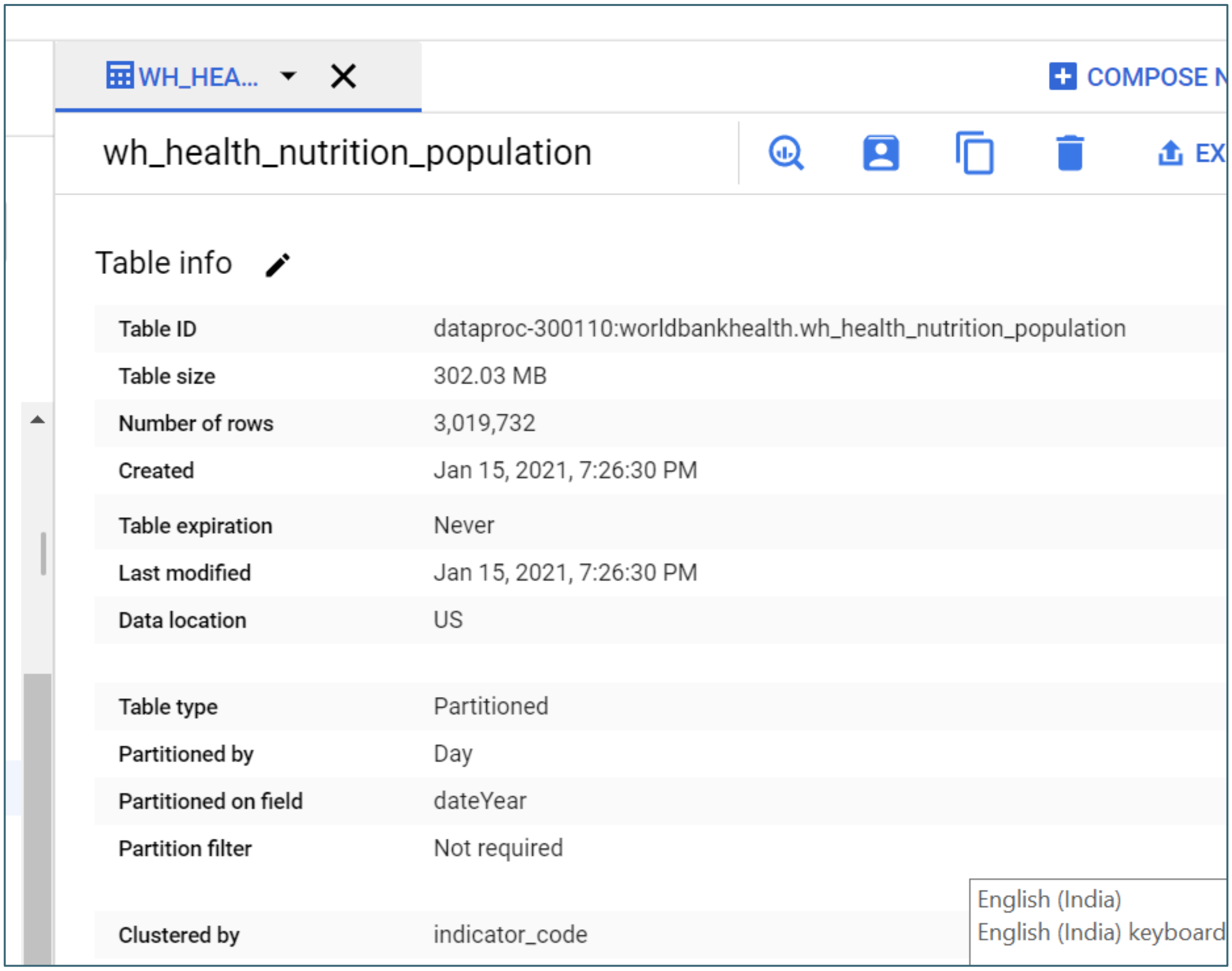

Figure 8: Table loaded - Partitioned and Clustered

Visualizations using Datalab

Visualizations were done using Datalab.

The following jupiter notebook was created and visualizations generated from the bigquery tables. Alternatively the jupiter notebook has been uploaded in the following location:

https://github.com/Sadiya-Dalvi/GoogleComposer/blob/main/GH-Project-Corr.ipynb

```
import seaborn as sn
import matplotlib.pyplot as plt
import pandas as pd

%load_ext google.cloud.bigquery
```

What is the average age of males and females in different countries around the world.

As there is no direct way to calculate average age, the below approach is taken:

We have indicator codes like 'SP.POP.%.MA.5Y' which give us % population between each 5Y band from age 0 to age 80+ example codes are SP.POP.10-14.MA.5Y , SP.POP.45-49.MA.5Y

Below query extracts the starting age of each 5Y block and adds 2.5 to it to get the average age We already have the % population in that age band as value field. This is added up for each country to get the average age in each country:

```
%%bigquery df_avg_age
SELECT round(sum(((cast(substr(indicator_code,8,2) as INT64) + 2.5) * value))/100,2) as Av
g_AGE ,  country_code, country_name
FROM `bigquery-public-data.world_bank_health_population.health_nutrition_population`
where indicator_code like 'SP.POP.%.MA.5Y' and year = 2019
group by country_code, country_name
order by Avg_AGE DESC
```

Top 20 countries with highest Average Ages

```
%%bigquery df_avg_age
SELECT round(sum(((cast(substr(indicator_code,8,2) as INT64) + 2.5) * value))/100,2) as Av
g_AGE ,  country_code, country_name
FROM `bigquery-public-data.world_bank_health_population.health_nutrition_population`
```

```
where indicator_code like 'SP.POP.%.MA.5Y' and year = 2019
group by country_code, country_name
order by Avg_AGE DESC limit 20
```

```
df_avg_age.plot(kind="bar", x="country_name", y="Avg_AGE")
<matplotlib.axes._subplots.AxesSubplot at 0x7faa79ecb358>
```

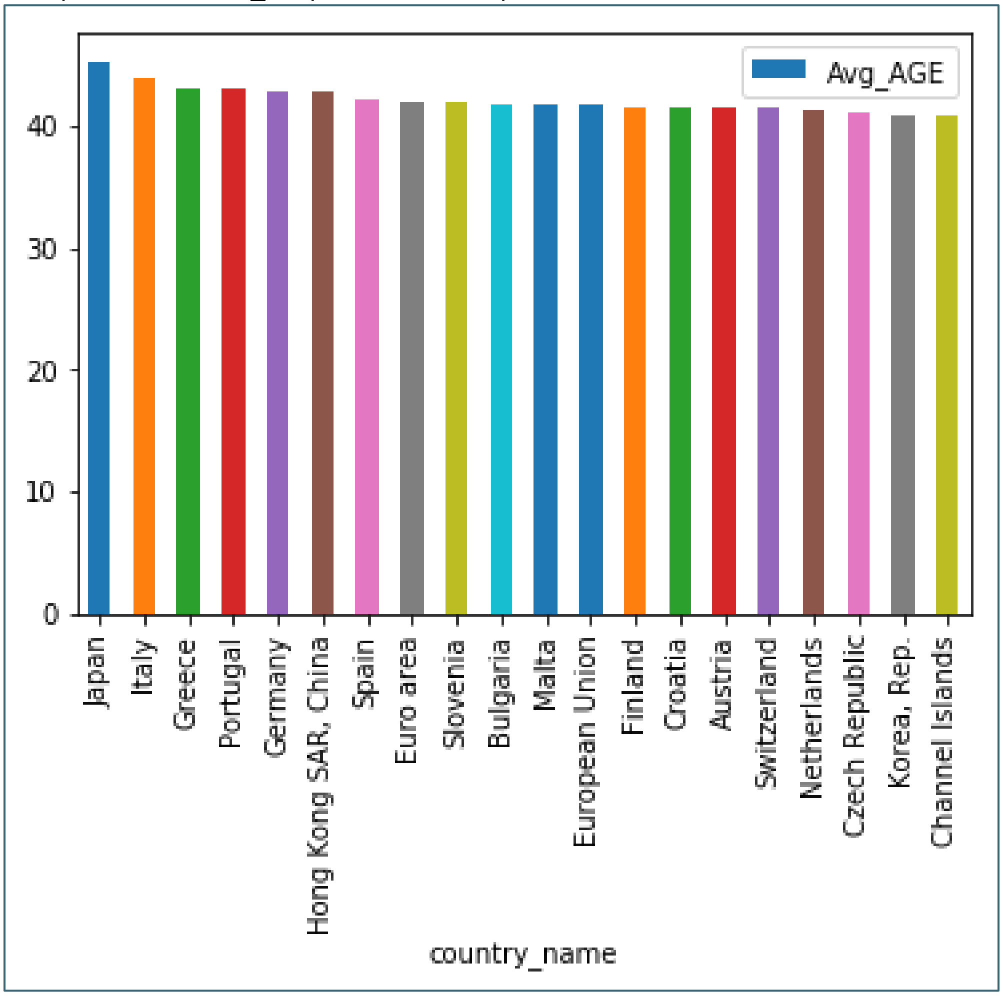

Figure 9: Highest Average Age across the world

```
df_avg_age.describe()
```

	Avg_AGE
count	20.000000
mean	42.175500
std	1.107099
min	40.890000

	Avg_AGE
25%	41.535000
50%	41.795000
75%	42.950000
max	45.350000

20 countries with lowest Average Ages

```
%%bigquery df_avg_age
SELECT round(sum(((cast(substr(indicator_code,8,2) as INT64) + 2.5) * value))/100,2) as Av
g_AGE ,  country_code, country_name
FROM `bigquery-public-data.world_bank_health_population.health_nutrition_population`
where indicator_code like 'SP.POP.%.MA.5Y' and year = 2019
group by country_code, country_name
order by Avg_AGE ASC limit 20
```

```
df_avg_age.plot(kind="bar", x="country_name", y="Avg_AGE")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7faa78adba58>
```

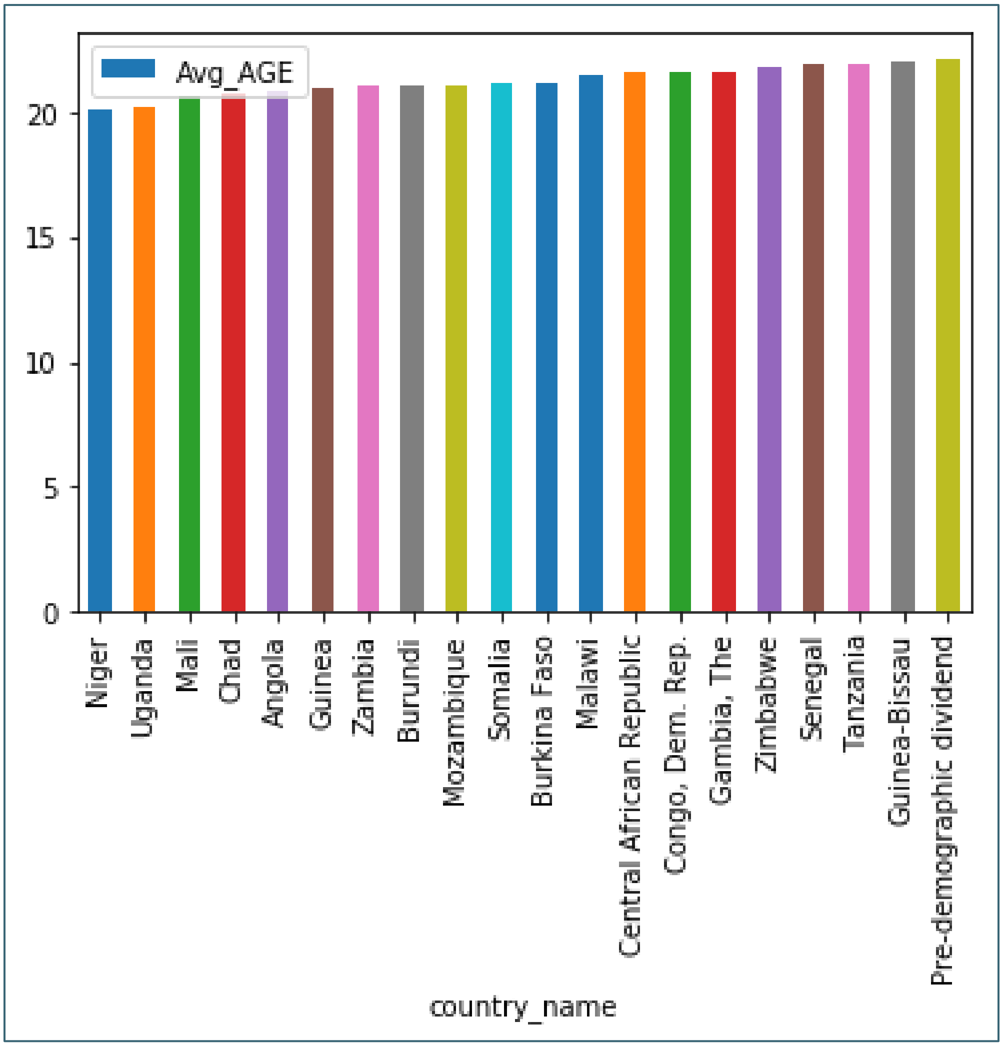

Figure 10: Lowest Average Ages across the world

Correlation between Health Expenditure and survival ages in the two genders around the world

```
%%bigquery survival_male
select  value as survival_male_65, country_name, year
from dataproc-300110.worldbankhealth.health_nutrition_population
where indicator_code in ('SP.DYN.TO65.MA.ZS') and year > 2000
```

```
%%bigquery health_exp
select value as health_exp, country_name, year
from dataproc-300110.worldbankhealth.health_nutrition_population
where indicator_code in ('SH.XPD.CHEX.PC.CD') and year > 2000
```

```
merge_pd = pd.merge(survival_male, health_exp, on=['country_name','year'])

corr_df = merge_pd

corr_df.drop(['year', 'country_name'], inplace=True, axis='columns')
corrMatrix = corr_df.corr()

sn.heatmap(corrMatrix,cmap='RdBu', center=0, annot=True)
plt.savefig('heathexp_survival_male_correlation.png')
plt.show()
```

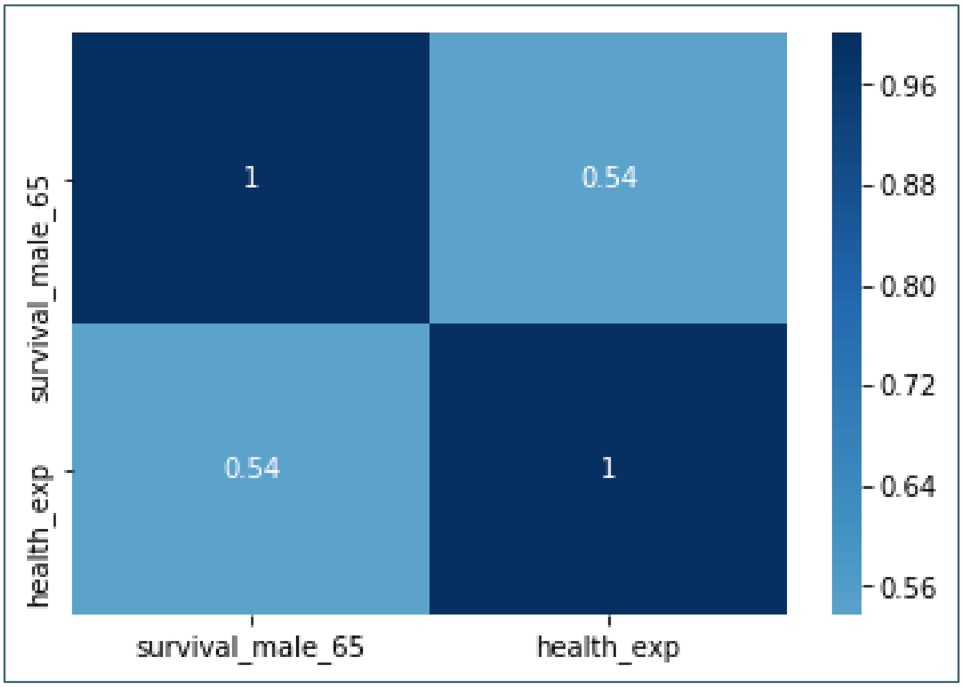

Figure 11: Correlation between Health Expenditure and Survival Ages of Males

```
%%bigquery survival_female
select  value as survival_female_65, country_name, year
from dataproc-300110.worldbankhealth.health_nutrition_population
where indicator_code in ('SP.DYN.TO65.FE.ZS') and year > 2000

merge_pd = pd.merge(survival_female, health_exp, on=['country_name','year'])
corr_df = merge_pd

corr_df.drop(['year', 'country_name'], inplace=True, axis='columns')
corrMatrix = corr_df.corr()

sn.heatmap(corrMatrix,cmap='RdBu', center=0, annot=True)
plt.savefig('heathexp_survival_female_correlation.png')
plt.show()
```

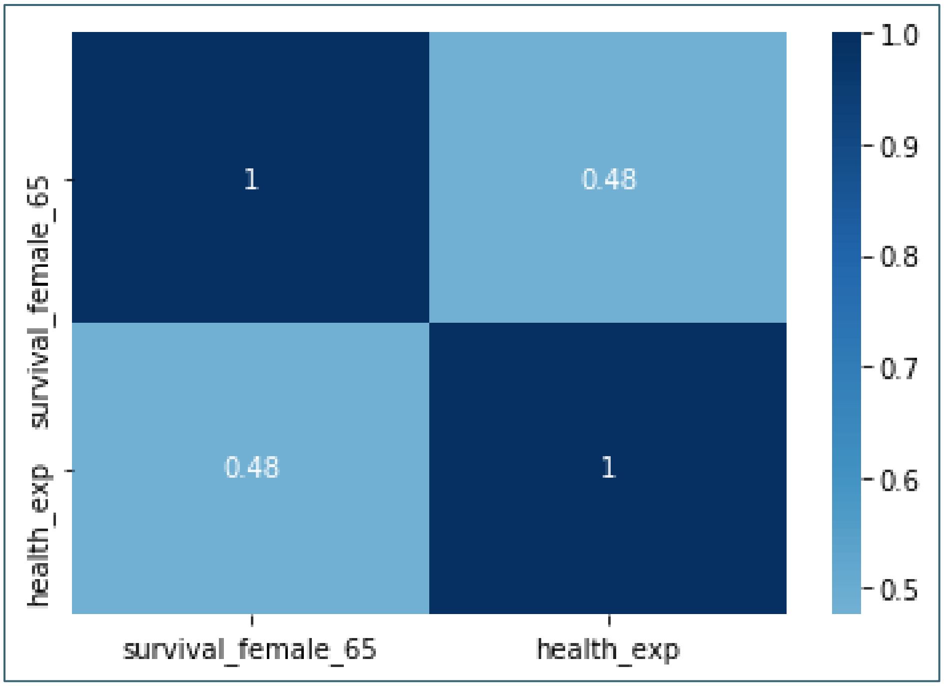

Figure 122: Correlation between Health Expenditure and Survival Ages of Females

Correlation between School enrollment and unemployment for males and females

School enrollment, secondary, male (% net) - SE.SEC.NENR.MA Unemployment, male (% of male labor force) - SL.UEM.TOTL.MA.ZS

```
%%bigquery df_school_enroll
select  value as school_enroll, country_name, year
from dataproc-300110.worldbankhealth.health_nutrition_population
where indicator_code in ('SE.SEC.NENR.MA') and year > 2000

%%bigquery df_unemploy
select value as unemploy, country_name, year
from dataproc-300110.worldbankhealth.health_nutrition_population
where indicator_code in ('SL.UEM.TOTL.MA.ZS') and year > 2000

merge_pd = pd.merge(df_school_enroll, df_unemploy, on=['country_name','year'])

corr_df = merge_pd

corr_df.drop(['year', 'country_name'], inplace=True, axis='columns')
corrMatrix = corr_df.corr()

sn.heatmap(corrMatrix,cmap='RdBu', center=0, annot=True)
```

```
plt.savefig('male_schoo_enroll_unemploy.png')
plt.show()
```

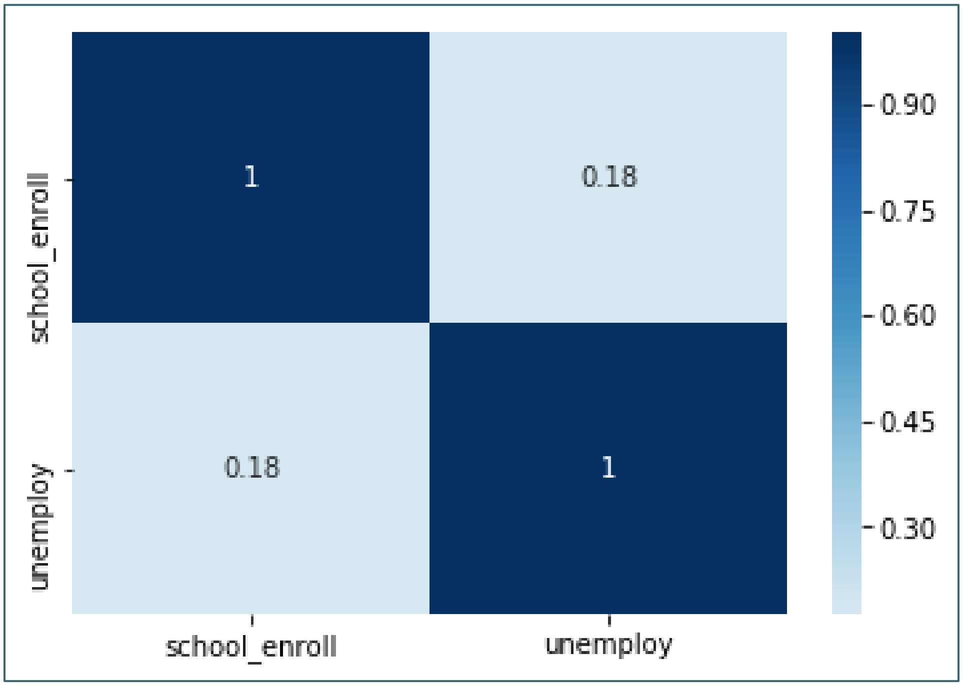

Figure 132: Correlation between School Enrollment and Unemployment of Males

Correlation between School enrollment and unemployment for females Using the below indicator code / indicator names

School enrollment, secondary, female (% net) - SE.SEC.NENR.FE Unemployment, female (% of female labor force) - SL.UEM.TOTL.FE.ZS

```
%%bigquery df_school_enroll
select  value as school_enroll, country_name, year
from dataproc-300110.worldbankhealth.health_nutrition_population
where indicator_code in ('SE.SEC.NENR.FE') and year > 2000
```

```
%%bigquery df_unemploy
select value as unemploy, country_name, year
from dataproc-300110.worldbankhealth.health_nutrition_population
where indicator_code in ('SL.UEM.TOTL.FE.ZS') and year > 2000
```

```
merge_pd = pd.merge(df_school_enroll, df_unemploy, on=['country_name','year'])

corr_df = merge_pd

corr_df.drop(['year', 'country_name'], inplace=True, axis='columns')
```

```
corrMatrix = corr_df.corr()

sn.heatmap(corrMatrix,cmap='RdBu', center=0, annot=True)
plt.savefig('male_schoo_enroll_unemploy.png')
plt.show()
```

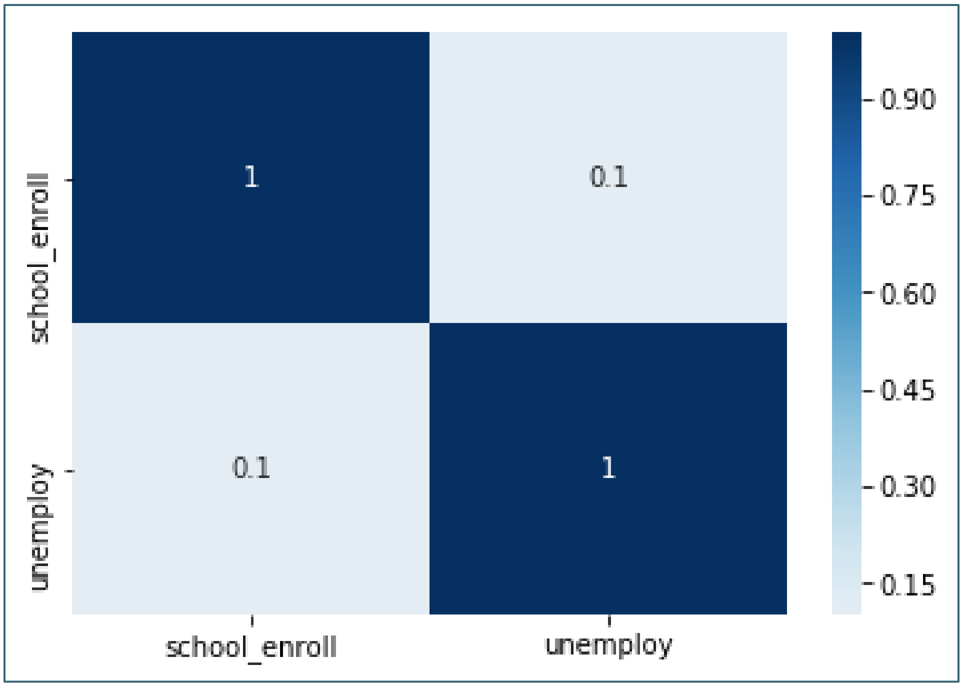

Figure 142: Correlation between School Enrollment and Unemployment of Females

Average age of first pregnancy around the world

Since the above indicator was not present in the dataset, the following assumptions are made:

Average age of first pregnancy is estimated as the average age of first marriage + 2. The indicator for age at first marriage, female - SP.DYN.SMAM.FE is available in the dataset

```
%%bigquery df_first_preg
SELECT round(value+2,1) as First_Pregnancy ,  country_code, country_name, year
FROM `dataproc-300110.worldbankhealth.wh_health_nutrition_population`
where indicator_code like 'SP.DYN.SMAM.FE' and year = 2013 and value > 15
order by value desc

df_first_preg.plot(kind="bar", x="country_name", y="First_Pregnancy")
<matplotlib.axes._subplots.AxesSubplot at 0x7faa78b49048>
```

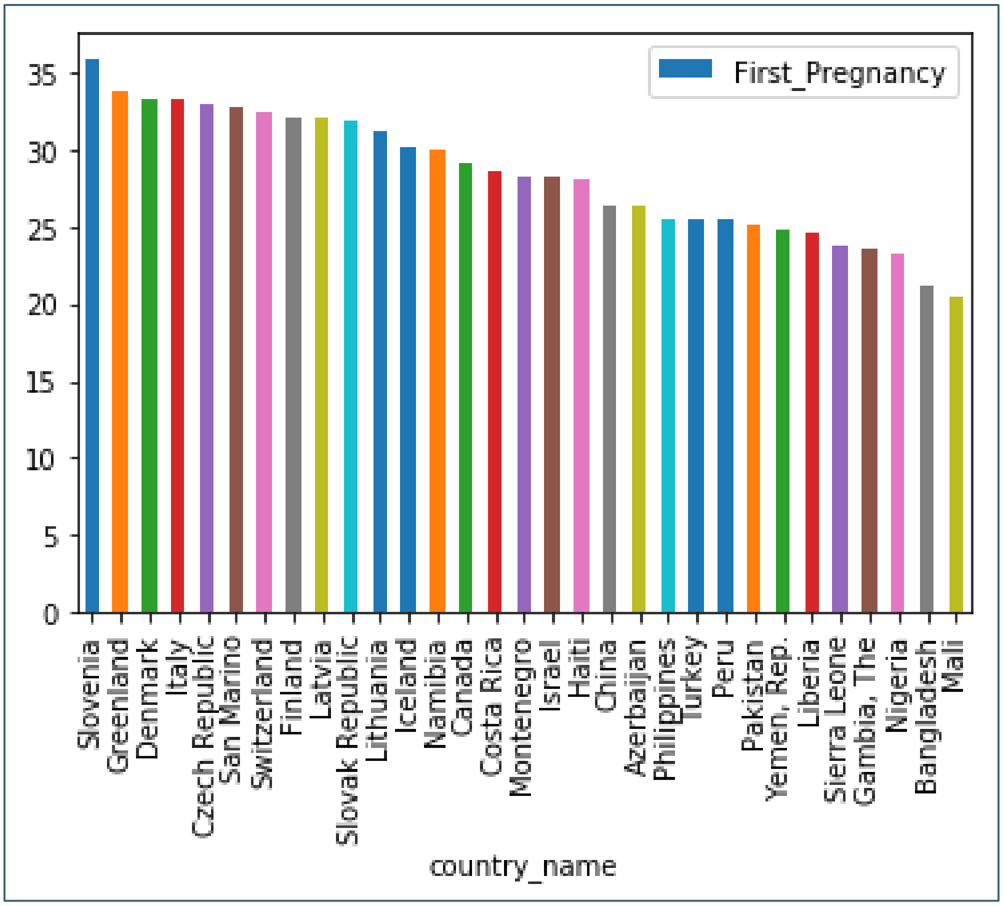

Figure 152: Average Age of First Pregnancy across the world

Analysis of age of marriage and infant mortality rate across different demographics

Mortality rate, infant (per 1,000 live births) – Indicator code is SP.DYN.IMRT.IN

Age of First marriage – Indicator code is 'SP.DYN.SMAM.FE'

```
%%bigquery df_mortality
SELECT round(value,1) as infant_mortality ,  country_code, country_name, year
FROM `dataproc-300110.worldbankhealth.wh_health_nutrition_population`
where indicator_code like 'SP.DYN.IMRT.IN' and year = 2013
order by value desc
```

```
df_mortality.describe()
```

	infant_mortality	year
count	234.000000	234.0

	infant_mortality	year
mean	25.038889	2013.0
std	21.528964	0.0
min	3.000000	2013.0
25%	6.725000	2013.0
50%	17.350000	2013.0
75%	38.375000	2013.0
max	96.600000	2013.0

```
%%bigquery df_first_marriage
SELECT round(value,1) as first_marriage ,  country_code, country_name, year
FROM `dataproc-300110.worldbankhealth.wh_health_nutrition_population`
where indicator_code like 'SP.DYN.SMAM.FE' and year = 2013 and value > 15
order by value desc
```

```
df_marriage_mortality = pd.merge(df_first_marriage, df_mortality, on=['country_name','year'])
```

```
df_marriage_mortality.plot(kind='bar',x='country_name',y=['first_marriage', 'infant_mortality'], secondary_y= 'infant_mortality')
```

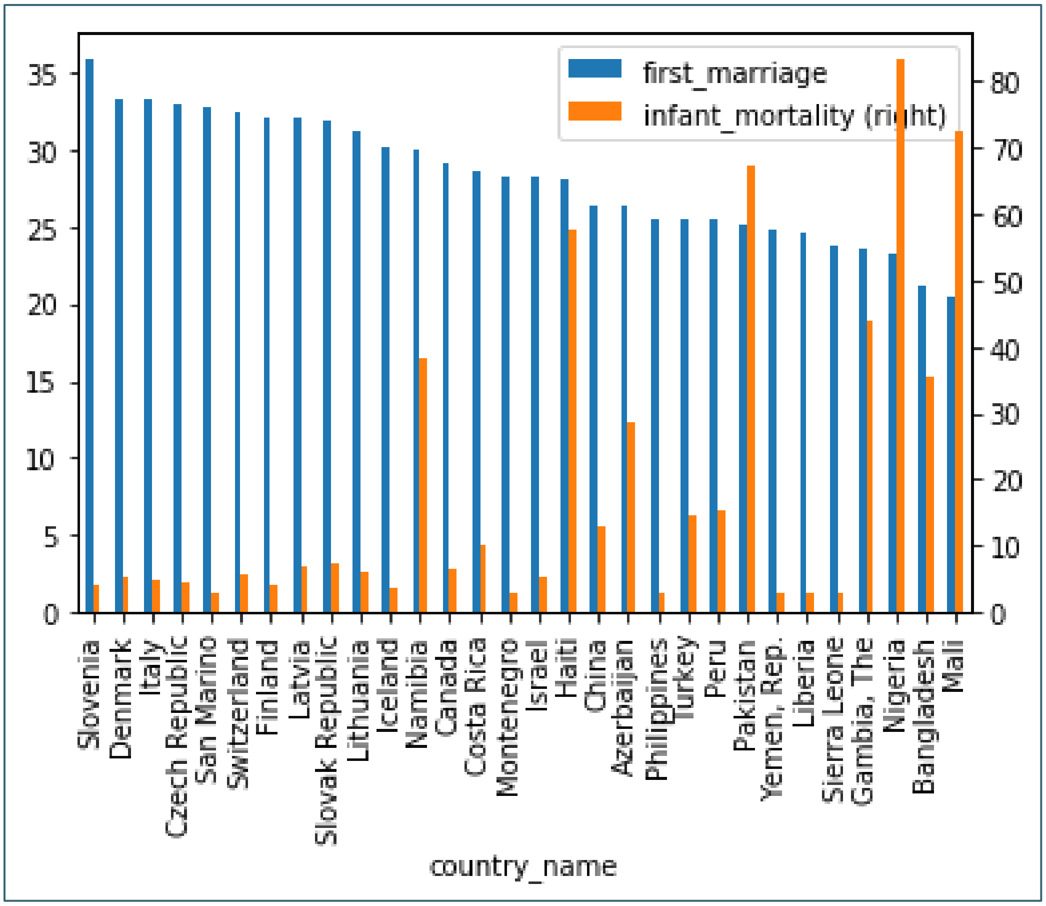

Figure 16: Age of marriage and infant mortality rate

3.7 Performance Comparison observed in BigQuery

In this section, we will compare the queries mentioned in the assignment with the raw dataset and the etl table.

Query	Raw Data Timings bigquery-public-data.world_bank_health_population.health_nutrition_population	Optimized ETL Table Timings dataproc-300110.worldbankhealth.wh_health_nutrition_population
Average age of males and females in different countries around the world		
`SELECT round(sum(((cast(substr(indicator_code,8,2) as INT64) + 2.5) * value))/100,2) as Avg_AGE , country_code, country_name FROM ` dataproc-300110.worldbankhealth.wh_health_nutrition_population ` where indicator_code like 'SP.POP.%.MA.5Y' and year = 2019 group by country_code , country_name order by Avg_AGE DESC`	Query complete (0.8 sec elapsed, 149.3 MB processed)	Query complete (0.8 sec elapsed, 2.4 MB processed)
Correlation between Health Expenditure and survival ages in the two genders around the world		
`select value as survival_male_65, country_name, year from dataproc-300110.worldbankhealth.wh_health_nutrition_population where indicator_code in ('SP.DYN.TO65.MA.ZS') and year > 2000`	Query complete (0.8 sec elapsed, 134.9 MB processed)	Query complete (0.5 sec elapsed, 52.8 MB processed)
`select value as health_exp, country_name, year from dataproc-300110.worldbankhealth.wh_health_nutrition_population where indicator_code in ('SH.XPD.CHEX.PC.CD') and year > 2000`	Query complete (0.4 sec elapsed, 134.9 MB processed)	Query complete (0.4 sec elapsed, 52.3 MB processed)

select value as survival_female_65, country_name, year from dataproc-300110.worldbankhealth.wh_health_nutrition_population where indicator_code in ('SP.DYN.TO65.FE.ZS') and year > 2000	Query complete (0.3 sec elapsed, 134.9 MB processed)	Query complete (0.3 sec elapsed, 52.8 MB processed)
Correlation between School enrollment and unemployment for males and females around the world		
select value as school_enroll, country_name, year from dataproc-300110.worldbankhealth.wh_health_nutrition_population where indicator_code in ('SE.SEC.NENR.MA') and year > 2000	Query complete (0.6 sec elapsed, 134.9 MB processed)	Query complete (0.4 sec elapsed, 50 MB processed)
select value as unemploy, country_name, year from dataproc-300110.worldbankhealth.wh_health_nutrition_population where indicator_code in ('SL.UEM.TOTL.MA.ZS') and year > 2000	Query complete (0.5 sec elapsed, 134.9 MB processed)	Query complete (0.3 sec elapsed, 52.3 MB processed)
select value as school_enroll, country_name, year from dataproc-300110.worldbankhealth.wh_health_nutrition_population where indicator_code in ('SE.SEC.NENR.FE') and year > 2000	Query complete (0.7 sec elapsed, 134.9 MB processed)	Query complete (0.4 sec elapsed, 50 MB processed)
select value as unemploy, country_name, year from dataproc-300110.worldbankhealth.wh_health_nutrition_population where indicator_code in ('SL.UEM.TOTL.FE.ZS') and year > 2000	Query complete (0.4 sec elapsed, 134.9 MB processed)	Query complete (0.3 sec elapsed, 52.3 MB processed)

Average age of first pregnancy around the world		
`SELECT round(value+2,1) as First_Pregnancy , country_code, country_name, year FROM ``dataproc-300110.worldbankhealth.wh_health_nutrition_population`` where indicator_code like 'SP.DYN.SMAM.FE' and year = 2013 and value > 15 order by value desc`	Query complete (0.3 sec elapsed, 149.3 MB processed)	Query complete (0.5 sec elapsed, 3.2 MB processed)
Analysis of age of marriage and infant mortality rate across different demograpics		
`SELECT round(value,1) as infant_mortality , country_code, country_name, year FROM ``dataproc-300110.worldbankhealth.wh_health_nutrition_population`` where indicator_code like 'SP.DYN.IMRT.IN' and year = 2013 order by value desc`	Query complete (0.4 sec elapsed, 149.3 MB processed)	Query complete (0.3 sec elapsed, 3.2 MB processed)
`SELECT round(value,1) as first_marriage , country_code, country_name, year FROM ``dataproc-300110.worldbankhealth.wh_health_nutrition_population`` where indicator_code like 'SP.DYN.SMAM.FE' and year = 2013 and value > 15 order by value desc`	Query complete (0.3 sec elapsed, 149.3 MB processed)	Query complete (0.2 sec elapsed, 3.2 MB processed)