The Life-Saving Mission for COVID-19 Vaccination on Google Cloud (GC) Ecosystem

Ramamurthy Valavandan[1], Kumaraswamy Reddy[2], Prasanth Parayatham[3], Ubaiyadulla Sherif[4], Pallav Kohli[5], Vikram Sharma[6], Pragathi S[7], Vijay R[8], Surasa Mukherjee[9], Nitin Ambekar[10], Dinesh Sai Teja Neeli[11], Santosh Baran[12], Vijender Singh[13], Saurabh Uniyal[14], Praveen B[15], Musheer Ahmed N[16]

[1, 2, 3, 44, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]Kyndryl Solutions Private Limited, GC Guild Project, 6th Floor, D-1, Manyata Tech Park, Bangalore 560024, Karnataka, India

[1]Corresponding Author Email: *ramamurthy.valavandan[at]kyndryl.com*
[2]*kumaraswamy.yanamala[at]kyndryl.com*
[3]*prasanth.parayatham[at]kyndryl.com*
[4]*ubaiyadulla.sherif[at]kyndryl.com*
[5]*pallav.kohli[at]kyndryl.com*
[6]*vikram.sharma1[at]kyndryl.com*
[7]*pragathi.s[at]kyndryl.com*
[8]*vijay.r1[at]kyndryl.com*
[9]*surasa.mukherjee[at]kyndryl.com*
[10]*nitin.ambekar[at]kyndryl.com*
[11]*dinesh.sai.teja.neeli[at]kyndryl.com*
[12]*santosh.baran[at]kyndryl.com*
[13]*vijender.singh[at]kyndryl.com*
[14]*saurabh.uniyal[at]kyndryl.com*
[15]*praveen.b[at]kyndryl.com*
[16]*musheer.ahmed.n[at]kyndryl.com*

Abstract: ***Google Cloud (GC) provisioning the ecosystem for stakeholders involved in the vaccination drive for World Health Organization (WHO). The COVID-19 vaccination dataset is available in the WHO portal and refreshed every day. In this paper, the pain area of collecting the source of data and the region or location of the data collection is addressed. A python program is developed to connect to the WHO portal with the help of Google Cloud Scheduler and process comma-separated variable (CSV) data on daily basis. The parameter of the location of data ingestion is parsed by Google Cloud Big Query by the data analytics and Log Analytics for capturing the location of the data ingestion. An ecosystem developed using Google Cloud Big Query for data analytics and Google Data Studio for data visualization is key for decision-makers of vaccination drive. In the process of access to the WHO dataset available in the WHO, the portal helps the researchers and stakeholders and visions the data visualization in the design and development. Also helps the researchers to give insight into the vaccination data and add value to the beneficiary.***

Keywords: Google Cloud, COVID- 19, Vaccination, Python, Data Visualization, WHO, Google data studio, Big Query, Data analytics, Cloud storage, Compute Engine

1. Introduction

Creating a Cloud ecosystem in Google Cloud for the public community, healthcare research, medical practitioners, government, and private body to access the Google data for the scope for decision making in the healthcare domain.

2. Google Cloud (GC) (Services and Resources)

2.1 GC for Data Analytics

Identification of GC services and resources for data and analytics Services.

2.2 GC Project

The project scope is to create a cloud ecosystem in Google Cloud (GC) as Infrastructure Modernization.

Functional scope: To create 'Use Cases' in Google Cloud for accessing COVID-19 vaccination dataset of World Health Organization (WHO).

Services used in the projects are Google Storage, Compute Engine, and ‘Big Query’ for Data Analytics.

Development: Client to store the files and provide the access on-demand in GC

2.3 SKUs | Google Cloud

https://cloud.google.com/skus

A Google Cloud Enterprise Agreement contracted directly by Google is eligible for the following Service Families, in addition to Google Cloud Services (availability of specific services may vary by country):

- Business Application Platform: Apigee and AppSheet
- Business Intelligence: Looker
- Compute Solutions: Bare Metal Solution

- Productivity Applications: Google Workspace subscription products
- Productivity Applications: Google Workspace usage-based products
- Security: Chronicle
- Security: Virus Total

A Google Cloud Enterprise Agreement contracted by Partners is eligible for the following Service Families, in addition to Google Cloud Services

- Business Application Platform: Apigee and AppSheet
- Business Intelligence: Looker
- Productivity Applications: Google Workspace subscription products
- Productivity Applications: Google Workspace usage-based products

2.2.1. GC-Big Query

GCData Analytics Service, Big Query is considered for processing the vaccination dataset of WHO. The process is created for patch jobs on getting the daily records uploaded to the WHO Portal.

2.2.2. Big Query External Table

The Big Query SQL is created for an external table dataset in the ‘tracking-matrix’. The syntax for an external table for the project.

```
CREATE
EXTERNAL
TABLE `tracing-matrix.covid19.WHO_Vaccination_data`
```

2.2.3. GCLOUD CLI (Gcloud Commands)

The gcloud command line is your gateway to manage and interact with the cloud and offers a variety of options to automatically parse and format the results. Here, we have used in this use case demonstrates gcloud commands together with python to extract the data and create the external table with the utility to auto-generate the formats for Google Cloud-Native Services.

```
gcloud auth activate-service-account
-- key-file “nature-labs-key. json”
```

3. Google GDK CLI

3.1. gcloud utilities in the project

In order to set the tracing-matrix as a project, in CLI the following command is performed.

```
gcloud config set project tracing-matrix
```

then, to view the config settings in the GC

```
gcloud config list
[accessibility]
screen_reader = False
[compute]
region = us-central1
[core]
account = nature-labs[at]tracing-
matrix.iam.gserviceaccount.com
```

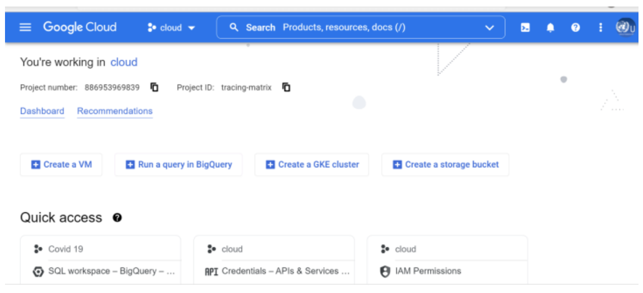


Figure 1: The caption of the GC Console

To create the ecosystem in Google Cloud and Dataset with details of the audit log of GC.
Source: WHO Coronavirus (COVID-19) data

Type of the data: Data is available in the comma-separated values (CSV) files

Research is carried out by the motivation of providing secure data ingestion, ecosystem, development of line of treatment, decision making in healthcare domain as per the WHO standards in healthcare.

3.2 Table (Schema and data integration)

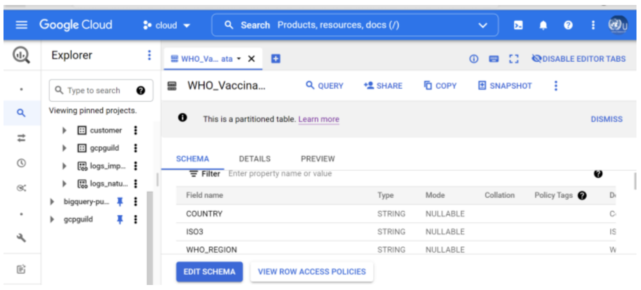
```
CREATE EXTERNAL TABLE `tracing-matrix. covid19.
WHO_Vaccination_data`
(
COUNTRY STRING,
ISO3 STRING,
WHO_REGION STRING,
DATA_SOURCE STRING,
DATE_UPDATED DATE,
TOTAL_VACCINATIONS FLOAT64,
PERSONS_VACCINATED_1PLUS_DOSE FLOAT64,
TOTAL_VACCINATIONS_PER100 FLOAT64,
PERSONS_VACCINATED_1PLUS_DOSE_PER100
FLOAT64,
PERSONS_FULLY_VACCINATED FLOAT64,
PERSONS_FULLY_VACCINATED_PER100 FLOAT64,
VACCINES_USED STRING,
FIRST_VACCINE_DATE DATE,
NUMBER_VACCINES_TYPES_USED FLOAT64,
PERSONS_BOOSTER_ADD_DOSE FLOAT64,
PERSONS_BOOSTER_ADD_DOSE_PER100 FLOAT64
)
OPTIONS (
skip_leading_rows=0,
format="CSV",
uris=
["https://drive.google.com/file/d/132PDmI2o9gParYa4F23o
_IqdR8Ncxo0J/view?usp=sharing"]
);
```

3.3. GC Compute Engine (Optimization)

To improve the big query to improve the performance of the GC, below _partitiontime for pseudo partitioning.

```
SELECT
*
FROM
`tracing-matrix.covid19.WHO_Vaccination data`
WHERE
```

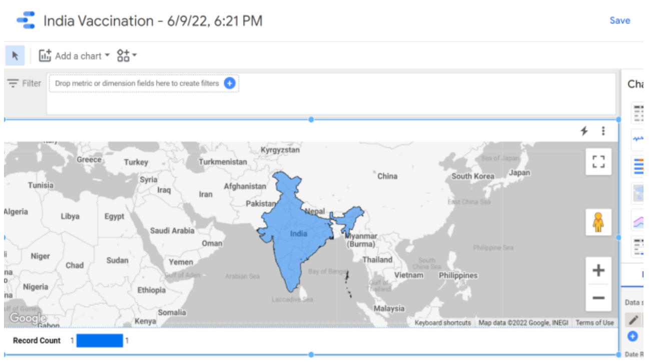
DATE (_PARTITIONTIME) = "2022-05-12"
AND
COUNTRY = 'India'

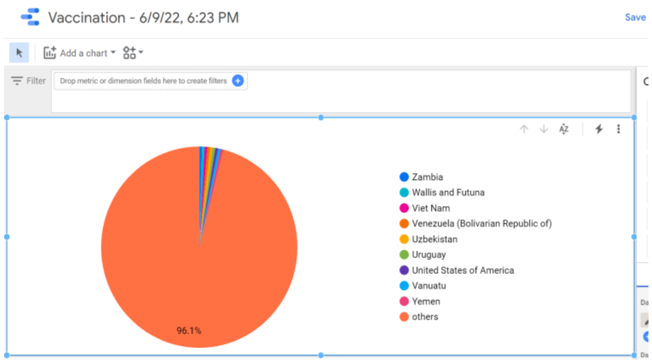


WHO_Vaccina...

This is a partitioned table. Learn more

Field name	Type	Mode	Collation	Policy Tags
COUNTRY	STRING	NULLABLE		
ISO3	STRING	NULLABLE		
WHO_REGION	STRING	NULLABLE		


India Vaccination - 6/9/22, 6:21 PM
Save
Add a chart
Filter
Drop metric or dimension fields here to create filters
India
Pakistan
China
Nepal
Myanmar (Burma)
Record Count



4. GC Log Analytics

GC log analytics helps to identify the location of the data ingestion.

```
SELECT
timestamp, resource. type, log_name, text_payload, proto_payload, json_payload
FROM
`logs_naturelabs_US._AllLogs`
WHERE
timestamp > TIMESTAMP_SUB (CURRENT_TIMESTAMP (), INTERVAL 100 DAY)
LIMIT 50
```

5. GC Project in Log Analytics

The project is designed and developed for the motivation of open source in healthcare. The project is a tax exception and noncommercial research program.

Project team: Google Cloud Guild Team, Kyndryl Solutions Private Limited.

Business requirement: Identification of sources of data and audit log of GC resources and Services.

Project outcome: Research papers and filing of a patent for GC Guild team.

A score of work: Create the ecosystem in Google Cloud for the public community, healthcare research, medical practitioners, government, and private body to access the Google data for the scope for decision making in the healthcare domain.

Purpose of the research: By publishing the sources of data ingestion from various sources papers and getting the insight of the COVID-19 data of WHO and development of Line of Treatment and collecting the data clenching and providing complete end-to-end ecosystem in Google Cloud (GC).

Task: To create the ecosystem in Google Cloud and Dataset with details of the audit log of GC.

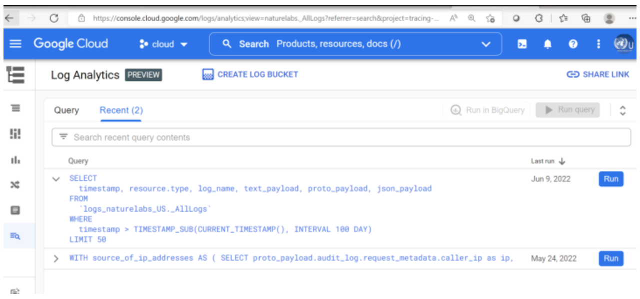
Source: WHO Coronavirus (COVID-19) data

Type of the data: Data is available in the comma-separated values (CSV) files

Research is carried out with the motivation of providing secure data ingestion, ecosystem, development of a line of treatment, and decision making in the healthcare domain as per the WHO standards in healthcare.

The log analytics query:

```
SELECT
STRUCT (proto_payload. type AS type,
STRUCT (proto_payload. audit_log.service_name AS service_name,
proto_payload.audit_log.method_name AS method_name,
proto_payload.audit_log.resource_name AS resource_name,
proto_payload.audit_log.resource_location AS resource_location,
proto_payload.audit_log.resource_original_state AS resource_original_state,
proto_payload.audit_log.num_response_items AS num_response_items,
proto_payload.audit_log.status AS status,
STRUCT (proto_payload.audit_log.authentication_info. principal_email AS principal_email,
proto_payload.audit_log.authentication_info. authority_selector AS authority_selector,
proto_payload.audit_log.authentication_info.third_party_principal AS third_party_principal,
proto_payload.audit_log.authentication_info. service_account_key_name AS service_account_key_name,
proto_payload.audit_log.authentication_info.service_account_delegation_info AS service_account_delegation_info,
proto_payload.audit_log.authentication_info. principal_subject AS principal_subject) AS authentication_info,
proto_payload.audit_log.authorization_info AS authorization_info,
proto_payload.audit_log.policy_violation_info AS policy_violation_info,
STRUCT
(proto_payload.audit_log.request_metadata.caller_ip AS
caller_ip,
proto_payload.audit_log.request_metadata.caller_supplied_user_agent AS caller_supplied_user_agent,
proto_payload.audit_log.request_metadata.caller_network
AS caller_network,
proto_payload.audit_log.request_metadata.request_attributes
AS request_attributes,
proto_payload.audit_log.request_metadata.destination_attributes AS destination_attributes) AS request_metadata,
proto_payload.audit_log.request AS request,
proto_payload.audit_log.response AS response,
proto_payload.audit_log.metadata AS metadata,
proto_payload.audit_log.service_data AS service_data) AS
audit_log,
proto_payload.request_log AS request_log) AS
proto_payload,
STRUCT (operation. id AS id,
operation.producer AS producer,
operation.first AS first,
operation.last AS last) AS operation
FROM
`logs_naturelabs_US._AllLogs` la
```


https://console.cloud.google.com/logs/analytics;view=naturelabs._AllLogs?referrer=search&project=tracing-...
Google Cloud
cloud
Search Products, resources, docs (/)
Log Analytics PREVIEW
CREATE LOG BUCKET
SHARE LINK
Query
Recent (2)
Run in BigQuery
Run query
Search recent query contents
Query
Last run
Run
Run

```
SELECT
  timestamp, resource.type, log_name, text_payload, proto_payload, json_payload
FROM
  `logs_naturelabs_US._AllLogs`
WHERE
  timestamp > TIMESTAMP_SUB(CURRENT_TIMESTAMP(), INTERVAL 100 DAY)
LIMIT 50
```
Jun 9, 2022

```
WITH source_of_ip_addresses AS ( SELECT proto_payload.audit_log.request_metadata.caller_ip as ip,
```
May 24, 2022

6. Life saving Services of GC

With the Big Query, all ethical approval and consent are taken with GC Big Query to provide the data analytics and Google Dashboard for the health care based on Blockchain and Privacy Computing).

GC has provided the analytical view and data ingestion of Vaccination from every country and locations. Google Log Analytics provides the information on a complete view of the vaccination dataset.

Vaccination used
Bharat-Covaxin,
Biological E-Corbevax,
Gamaleya-Gam-Covid-Vac,
Janssen-Ad26.COV 2-S,
Moderna-Spikevax,
SII-Covishield, SII-Covovax,
Zydus-ZyCov-D

The cumulative records of Vaccination

Country	Total_Vaccinations	Persons_Vaccinated_1plus_Dose
India	1906551885	1004921253

Nature Labs is the United Nations research wing in COVID-19 and has involved Kyndryl Solutions Private Limited for the dataset creation, python, Big Query, Compute Engine.

caller_ip	count
122.161.50.5	429
106.201.116.76	296
106.201.116.76	196
49.204.201.54	160
202.12.83.129	117
202.164.136.44	106
115.96.183.233	65
103.99.109.66	50
49.204.135.142	17
223.187.115.248	15
223.187.123.202	6
27.34.241.129	5
202.164.136.44	5

6.1 GC Services and resources

GC Guild has followed WHO healthcare as it is a health care based on Blockchain and Privacy Computing.

Google Cloud (GC), big query is allowed by the Kyndryl Solutions Private Limited in WHO dataset in the cloud platform. All the authors are pleased to support the publishers and any further communication from the readers and stakeholders.

GC Service	GC References
Compute Engine	a. https://cloud.google.com/compute
	b. particular Managed Instance Groups for scaling
	https://cloud.google.com/compute/docs/instance-groups#managed_instance_groups
IaC:use tools like Terraform to create multiple environments:	https://cloud.google.com/docs/terraform
Cloud CDN can provide Content Delivery Network services	https://cloud.google.com/cdn
Google Workspace can provide email services and plenty more when it comes to employee collaboration	https://workspace.google.com/
MariaDB SkySQL runs on Google Cloud	a. https://mariadb.com/products/skysql/google-cloud-platform/
	b. Or you can have Microsoft SQL Server, MySQL and PostgreSQL as a service through the Cloud SQL service: https://cloud.google.com/sql
For high volume, high performance storage for assets, nothing better than Google Cloud Storage	https://cloud.google.com/storage
And for shared NFS storage for web servers, check out Filestore	https://cloud.google.com/filestore
One can reserve public static IP addresses for web applications-be it a VM or a load	https://cloud.google.com/compute/docs/ip-addresses/reserve-static-external-ip-address

balancer	
Backup and Disaster Recovery tooling available	https://cloud.google.com/solutions/backup-dr
A broad range of SLAs are available and depending on solution architecture	a. https://cloud.google.com/terms/sla there is also comprehensive support offering depending on need
	b. https://cloud.google.com/support

6.2 Python for use cases in vaccination data

The dataset is available for 24X7X365 days as per the Google Platform (GC) provisioning in the cloud service level agreement.

```
"""
Client: WHO-Nature Labs
Project Scope: GC as Infrastructure Modernization
Functional scope: Function to create 'Use Cases' in Google Cloud
GC Projects Used: Google Storage, Compute Engine
Development: Client to storage the files and provide the access on-demand in GC

Generate the Google Storage and Generate the compute engine for performance
Written by Kyndryl for GC Data store location in Nature Labs Project
Author: GCguild@gmail.com
gcloud components
"""
ipfile="vaccination-data.csv"
#ifl="Latest reported counts of cases and deaths"

currentdirds="ds3"
coviddir="covid19"
basepath = "C:\\nature-labs\\who"
gcli="WHO"

#C:\nature-labs\who\covid19\ds3
projectID="tracing-matrix"
dataset="covid19"

URI="https://drive.google.com/file/d/132PDmI2o9gParYa4F23o_IqdR8Ncxo0J/view?usp=sharing"

ifc="Vaccination data"

sl=27
ls=len (ifc)
if (ls <= sl):
 ifl=ifc
else:
 ifl= (ifc [0:sl])

"""
NO CHANGE SHOULD BE DONE AFTERWARDS. . .
"""

gcloudcodepaths = ("{}{}{}{}{}". format (basepath, "\\", coviddir, "\\", currentdirds))

chkwho = ("{}{}{}". format (gcloudcodepaths, "\\", ipfile))
import re
import glob
from tkinter import W
import pandas as pd

from pandas import ExcelWriter
from pandas import ExcelFile
from os. path import expanduser as ospath

from pathlib import Path
import logging
import socket
from inspect import getsourcefile

import chardet
import pandas as pd

from datetime import datetime

import shutil
import xlrd

import runpy

import os
import sys

logf ="GClog. txt"
logfi = ("{}{}". format ("\\", logf))
logfile = (gcloudcodepaths + logfi)

logging. basicConfig (
 filename = logfile,
 level = logging.INFO,
 format = '% (levelname) s: % (asctime) s: % (message) s')

logging. info ('Compute Engine Directory: %r', {gcloudcodepaths})

fileinfo= (os.path.split (sys.argv [0]) [1])
hostname= (socket.gethostbyaddr (socket. gethostname ()) [0])
datestamp = datetime.now ().date ()

logging.info ('------ Start of Google Log Analytics Projects---------')
logging.info ('Host Name %r, Compute Engine = %r', hostname, fileinfo)

path = Path (chkwho)

def prt (p):

width = len (p) + 4
print (' ┏' + "━"*width + "┓")
print (' ┃' + p. center (width) + ' ┃')
print (' ┗' + "━"*width + "┛")

if path. is_file ():
 pi="\'Excel file is created \': "
 p = ("{} {}". format (pi, chkwho))
 prt (p)
```

```
else:
 pi="\'excelfilefordtye is missing !\': "
 p = ("{} {}". format (pi, chkwho))
 prt (p)
 logging. error ('Could not find xls file: %r', {p})
 exit (1)

targetdir = ("{}{}{}". format (basepath, "\\", currentdirds))

sno= ("{}_{}". format (gcli, ifc))

dc=sno
dc = re.sub (' [^A-Za-z0-9]+', ' ', dc)
dc = dc.strip ()
dc = dc.rstrip ()
dc = dc.lstrip ()
dc = re.sub ("\s", "_", dc)

N="\\"
csvout = ("{}{}. {}". format (N, dc, "csv"))

pi = "\'Before Renaming \': "
p = ("{}\t{}". format (pi, chkwho))

prt (p)

csvfileforuploadcsv = (gcloudcodepaths + csvout)

pi="\'After Renaming \': "
p = ("{}\t{}". format (pi, csvfileforuploadcsv))
prt (p)

shutil. copy (chkwho, csvfileforuploadcsv)

sno= ("{}_{}". format (gcli, ifl))
fno= ("{}_{}". format (gcli, ifc))

filetylst= ['sql', 'csv', 'xlsx']
dc=fno
dc = re.sub (' [^A-Za-z0-9]+', ' ', dc)
dc = dc.strip ()
dc = dc.rstrip ()
dc = dc.lstrip ()
dc = re.sub ("\s", "_", dc)
ext_table_name=dc
N="\\"
for fl in (filetylst):
 thr= ("{}. {}". format (dc, fl))
 if (fl == 'sql'):
 bqf = ("{}{}{}". format (N, 'BQ_', thr))
 if (fl== 'csv'):
 incsv = ("{}{}". format (N, thr))
 csvout = ("{}{}{}{}". format (N, 'Upload_', 'GC_', thr))
 else:
 outxls = ("{}{}". format (N, thr))

infc = (gcloudcodepaths + incsv)
conxls = (gcloudcodepaths + outxls)
csvfileforupload = (gcloudcodepaths + csvout)
bqfile = (gcloudcodepaths + bqf)
excelfilefordtye=csvfileforupload

with open (infc, 'rb') as f:
 enc = chardet.detect (f.read ())
dfc = pd.read_csv (infc, encoding = enc ['encoding'])

dfc. to_excel (conxls, sheet_name=sno, index=False)


excelfilefordtye = conxls

xl = pd.ExcelFile (excelfilefordtye)

sheetlst=xl.sheet_names

for sn in (sheetlst):
 sheetname=sn
 logging. info ('Sheet Name: %r', sheetname)

with open (excelfilefordtye, "rb") as f:
 df_input_file = pd.read_excel (f, sheet_name=sheetname,
header=0, index_col=None)

colname=df_input_file. columns
datatypes=dict (df_input_file.dtypes)

row_count=df_input_file.count () [0]

logging.info ('No of rows in Input File %r, Row Count %r',
excelfilefordtye, row_count)

logging.info ('\n Generating the Project account in GC:\n')

logging.info ('Google Log Analytics Projects Generated File
%r', conxls)

df_input_file.head (row_count).to_csv (csvfileforupload,
encoding='utf-8', header=False, index=False)

def switch (check_data_type):
 dict={
 'object': 'STRING',
 'int64': 'INT64',
 'float64': 'FLOAT64',
 'DATE': 'DATE'
 }
 return dict.get (check_data_type, 'Unable to find Data
Type')

datearray= ['date', 'DATE', 'Date']
fldnames= []
for fld in colname:

 for cdatesrt in (datearray):
 check_date_return = fld.find (cdatesrt)
 check_date_lu=cdatesrt
 if (check_date_return !=-1):
 check_data_type='DATE'
 break

 else:
 dtdef=df_input_file [fld].dtypes
 check_data_type = str (dtdef)
 logging.info ('Field Name %r, Check Data Type %r, Check
DATE Return code %r', {fld}, {check_data_type},
{check_date_return})
```

```
 flddty=switch (check_data_type)
 pi="\'Check Data Type of Field is Date: \': "
 p = ("{}: {}: {}". format (pi, fld, flddty))

 dc=fld

 dc = re.sub (' [^A-Za-z0-9]+', ' ', dc)
 dc = dc.strip ()
 dc = dc.rstrip ()
 dc = dc.lstrip ()
 dc = re.sub ("\s", "_", dc)
 dc = re.sub (r" [^\w\s]", ", dc)
 dc = re.sub (r"\s+", '_', dc)
 ddc=dc
 logging.info ('Field Name: %r, Data Type: %r ', dc, flddty)
 logging.info ('Fld Name: %r, Original DTy %r: Converted
DTy is: %r', ddc, check_data_type, flddty)
 tblsting= ("{} {}". format (dc, flddty))
 fldnames.append (tblsting)

logging.info ('Elements in Table Field and Datatype %r',
fldnames)
L= []
lc=1
ll=len (fldnames)
logging.info ('Number of Elements in Tbl Fld and Dty List
or Array %r', ll)

for fldy in (fldnames):
 logging.info ('Field and DTy: %r ', fldy)
 fldy= ("{}\t{}". format ("\t", fldy))
 L.append (fldy)
 if (lc == ll):
 N="\n"
 else:
 N=", \n"
 lc += 1
 L.append (N)

cene = open (bqfile, 'w')

tbe=ext_table_name
tbe= re.sub (' [^A-Za-z0-9]+', ' ', tbe)
tbe = tbe.strip ()
tbe = tbe.rstrip ()
tbe = tbe.lstrip ()
tbe = re.sub ("\s", "_", tbe)
tbe = re.sub (r" [^\w\s]", ", tbe)
tbe = re.sub (r"\s+", '_', tbe)

ts= ("{} {} {}". format ("-- Generated schema for table: ",
tbe, "-- "))
fulltblname= ("{}. {}. {}". format (projectID, dataset, tbe))
line1= ("{} {}{}{}". format ("CREATE EXTERNAL
TABLE", "`", fulltblname, "`\n"))

line2=" (\n"
line3="\n)"
s = """
OPTIONS (
skip_leading_rows=0,
format="CSV",
"""
uris= ("{}{}{}{}". format ("uris= [", "\"", URI, "\"]"))
line4="\n);"
cene.write (line1)
cene.write (line2)
cene.writelines (L)

cene.write (line3)

cene.write (s)

cene.write (uris)

cene.write (line4)

cene.close ()

logformatfile = ("{}{}_{}_{}". format ("\\", hostname,
datestamp, logf))
logdfilenew = (gcloudcodepaths + logformatfile)

shutil.copy (logfile, logdfilenew)

postscript="cleanfiles.py"
cfls = ("{}{}". format ("\\", postscript))
cleanfile = (gcloudcodepaths + cfls)
clean = open (cleanfile, 'w')

def cleanfl (rmv, removefile):



 ldc = str (removefile)
 slfs = (ldc.split ('\\'))

 leba=len (slfs)-1
 for rf in range (0, len (slfs)):

 if (rf == 0):
 sla= ("{}{}". format (rmv, " = \""))

 else:
 sla= ("{}{}". format ("\\", "\\"))
 arf= ("{}{}". format (sla, slfs [rf]))
 larys.append (arf)
 if (rf == leba):
 dq= ("{}{}". format ("\"", "\n"))
 larys.append (dq)
 clean.writelines (larys)

removefile=logfile
larys= []
cleanfl ('logfile_rm', removefile)

larys= []
cleanfl ('excel_rm', excelfilefordtye)

s = """
import shutil
import os
import sys

#remove file if exists
def remove_if_exists (removefile):
 try:
 if os.path. exists (removefile):
 os.remove (removefile)
 print ("File removed successfully", removefile)
 except:
 print ("Error while deleting file ", removefile)

#remove previous log file

removefile = logfile_rm
remove_if_exists (removefile)

removefile = excel_rm
remove_if_exists (removefile)
"""
clean.write (s)

clean.close

pi="\'Create 'Table' GC Big Query \': "
p = ("{} {}". format (pi, bqfile))
prt (p)

pi="Execute 'python' for cleaning file (s): "
p = ("{} {}". format (pi, cleanfile))
prt (p)

pi="python "
p = ("{} {}". format (pi, cleanfile))
prt (p)
```

6.3 Competing interests

The data, program, and artifacts are available for all the stakeholders free of cost and there is no commercial interest.

6.4 Funding

GC Guild Team has contributed to the WHO dataset and created the program for the benefit of the COVID-19 vaccination and WHO – Nature Labs has provided the GC billing and support for the programming, big query knowledge transfer.

6.5 Authors' contributions

Kyndryl Solutions Private Limited, GC Guild members have contributed their time and efforts for the WHO successfully in provisioning the dataset and automated Python programming, Big Query Tables.

6.6 Python programming

WHO – Nature Labs research team acknowledged the contributions of Kyndryl Solutions Private Limited in creating the GC projects and providing the data migration, Big Query and Python programming for the data analytics. Also thankful to Google for providing the Log Analytics for Nature Labs project.

```
import requests
import re
import shutil
import os
import sys
who_data_url = 'https://covid19.who.int/who-data/vaccination-data.csv'
whodata=re.sub (r'^. +/ ([^/]+) $', r'\1', who_data_url)
workingdirctory="ds3"
customerdirctory="covid19"
basepath = "C:\\nature-labs\\who"
gcloudcodepaths = ("{}{}{}{}{}". format (basepath, "\\", customerdirctory, "\\", workingdirctory))

fullyqualifiedwhodata = ("{}{}{}". format (gcloudcodepaths, "\\", whodata))

def prt (p):

 width = len (p) + 4
 print ('┏' + "━"*width + "┓")
 print ('┃' + p. center (width) + '┃')
 print ('┗' + "━"*width + "┛")

#remove file if exists
def remove_if_exists (removefile):
 try:
 if os.path. exists (removefile):
 os.remove (removefile)
 #print ("File removed successfully", removefile)
 pi="\'File removed successfully \': "
 p = ("{}{}". format (pi, removefile))
 prt (p)
 except:
 print ("Error while deleting file ", removefile)

#remove previous log file

removefile = fullyqualifiedwhodata
remove_if_exists (removefile)

pi="\'Downloading WHO Vaccination data \': "
p = ("{}{}". format (pi, who_data_url))
prt (p)

def downloading (download_url, local_file_data):
 file_stream = requests. get (download_url, stream=True)
 with open (local_file_data, 'wb') as local_file:
 for data in file_stream:
 local_file.write (data)

download_url=who_data_url
local_file_data=fullyqualifiedwhodata
downloading (download_url, local_file_data)

pi="\'Download is completed: \': "
p = ("{}{}". format (pi, fullyqualifiedwhodata))
prt (p)
```

6.7 Authors' information (GC Guild)

Kyndryl Solutions Private Limited Authors are Subject Matter Expert, GC Cloud Platform Architect, Big Data

Engineering with solution and development experts for WHO – Nature Labs Project.

Acknowledgements

GC Guild team develops the Vaccination dataset for beneficiaries, World Health Organization (WHO) research operation, Nature Labs (United Nations Body) is responsible for the interpretation and use of the content lies with the reader. WHO reserves the right to make updates and changes to posted content without notice and accepts no liability for any errors or omissions in this regard. WHO assumes no responsibility or liability for any consequence resulting directly or indirectly from any action or inaction readers take based on or made in reliance on the information and material available on the WHO Health Emergency Dashboard. While every reasonable effort has been made to use appropriate language and pictures on the WHO Health Emergency Dashboard, WHO expressly disclaims any responsibility for inadvertent offensive or insensitive, perceived or actual, language or pictures. WHO will take no responsibility for or be liable for the WHO Health Emergency Dashboard being temporarily unavailable in the event of technical or other issues. The World Health Organization (WHO) Health Emergency Dashboard is a platform which aims to share information about public health events and emergencies. The data on the dashboard is refreshed every fifteen (15) minutes and data is accurate as at time of refreshing.

The WHO Health Emergency Dashboard is not a comprehensive representation of all the events and emergencies that WHO is aware of and responding to. The events displayed are a subset of those reported through official channels as mandated by the International Health Regulations (IHR 2005). The content of the WHO Health Emergency Dashboard is for general information only. It is subject to change without notice. While every reasonable effort has been made to make the information on the WHO Health Emergency Dashboard as timely and accurate as possible, WHO makes no claims, promises or guarantees about the effectiveness, completeness and accuracy of the contents of the WHO Health Emergency Dashboard, and expressly disclaims any liability for damages as a result of the use and/or application of the WHO Health Emergency Dashboard, errors and/or omissions in the content. The size of the pie charts corresponds to the number of events in a given country or territory; the size does not indicate the severity or risk associated with the event (s). The designations employed and the presentation of content on the WHO Health Emergency Dashboard, including names of the events, maps and other illustrative materials, do not imply the expression of any opinion whatsoever on the part of WHO concerning the legal status of any country, territory, city or area, or of its authorities, or concerning the delineation of frontiers and borders. Grey areas on maps represent approximate border lines for which there may not yet be full agreement.

References

[1] GC All products
[2] Google Cloud and services
[3] Management
[4] identity and access management, and use APIs

Name	Description
APIs & Services	API management for cloud services
Billing	Assortment of billing and cost management tools
IAM & Admin	Resource access control

[5] Compute
[6] Run scalable virtual machines and containers

Name	Description
Compute Engine	VMs, GPUs, TPUs, disks
Kubernetes Engine	Managed Kubernetes / containers
VMware Engine	VMware as a service
Anthos	Enterprise hybrid multi-cloud platform
Distributed Cloud	Managed edge infrastructure

[7] Storage
[8] Store long & term, VM, and Filestore securely

Name	Description
Cloud Storage	Enterprise-ready object storage
Filestore	Fully managed NFS server
Data Transfer	Secure and flexible way to move data

[9] Analytics
[10] Collect, store, process, and analyze large data

Name	Description
BigQuery	Data warehouse/analytics
Pub/Sub	Global real-time messaging
Dataflow	Streaming analytics service
Composer	Managed workflow orchestration service
Dataproc	Managed Apache Hadoop
Dataprep	Visual data wrangling
IoT Core	Device management and data ingestion
Data Fusion	Data pipeline management
Looker	Enterprise BI and Analytics
Healthcare	Healthcare data storage and processing
Financial Services	Revenue growth through the cloud
Datastream	Managed CDC service
Life Sciences	Biomedical data at scale
Data Catalog	Metadata management service
Elastic Cloud	Data power to uncover actionable intelligence
Databricks	Platform for data, analytics, and AI workloads

[11] Networking
[12] Manage, connect, secure, and scale your networks

Name	Description
VPC network	Virtual private cloud
Network services	Network management tools
Hybrid Connectivity	Network connectivity options
Network Security	Tools that power safe networking
Network Intelligence	Network monitoring and topology
Network Service Tiers	Price vs performance tiering

[13] Serverless
[14] Build applications powered by serverless functions and containers

Name	Description
Cloud Run	Serverless for containerized applications
Cloud Functions	Event-driven serverless functions
App Engine	Managed app platform

[15] Databases
[16] Create, manage, and migrate relational and non-relational databases

Name	Description
SQL	Managed MySQL, PostgreSQL, SQL Server
Datastore	NoSQL database for your web and mobile apps
Firestore	Serverless NoSQL document DB
Spanner	Horizontally scalable relational DB
Bigtable	Petabyte-scale, low-latency, non-relational
Memorystore	Managed Redis and Memcached
Database Migration	Cloud SQL migrations simplified
MongoDB Atlas	JSON-like data models, querying, & scaling
Neo4j Aura Professional Database-as-a-Service	Integrated, fully managed graph databases
Redis Enterprise	Robust in-memory database platform
DataStax Astra	Cloud-native Cassandra app development

[17] Operations
[18] Monitor, troubleshoot, and improve application performance

Name	Description
Logging	Real-time log management and analysis
Monitoring	Infrastructure and application quality checks
Error Reporting	Dashboard for app errors
Trace	App latency insights
Profiler	CPU and heap profiling
Debugger	Code investigation in production

[19] Security
[20] Meet policy and compliance objectives

Name	Description
Security	Security management controls and capabilities
Compliance	Tools that help support data regulations

[21] CI/CD
[22] Integrate and deliver continuously

Name	Description
Cloud Build	Continuous integration delivery platform
Container Registry	Private container registry storage
Source Repositories	Hosted private git repos
Artifact Registry	Universal build artifact management
Cloud Deploy	Managed continuous delivery to GKE

[23] Artificial Intelligence
[24] Leverage machine learning products on a trusted platform

Name	Description
Vertex AI	One AI platform, every ML tool you need
Vision	Custom image models, now also in Vertex AI
Natural Language	Custom text models
Tables	Custom data models, now also in Vertex AI
Translation	Language detection and translation
Document AI	Document analysis, classification, and searches
Recommendations AI	Custom recommendations for products
Video Intelligence	ML video analysis, now also in Vertex AI
Retail	Data-driven solutions for retailers
Data Labeling	Data labeling by humans
Speech-to-Text	Audio-to-text conversion
Talent Solution	Job search with ML

[25] Application Integration
[26] Enable applications and microservices to work together seamlessly

Name	Description
Cloud Scheduler	Managed cron job service
Cloud Tasks	Asynchronous task execution
API Gateway	API development, deployment, and management
Workflows	HTTP services orchestration
Eventarc	Modern event delivery

[27] Tools
[28] Discover productivity resources for your cloud

Name	Description
Endpoints	Cloud API gateway
Identity Platform	Google-grade identity and access management
Deployment Manager	Templated infrastructure deployment
Apigee	API Management
Service Catalog	Internal solutions catalog
Carbon Footprint	Your cloud carbon emissions
Apache Kafka on Confluent	Cloud-native managed service for Apache Kafka
Splunk Cloud	Data-to-Value Platform

[29] Other Google products
[30] Explore gaming and Google Maps Platform products

Name	Description
Google Maps Platform	Real-world insights and location experiences
Game Servers	Agones cluster orchestration

[31] Support
[32] Find live support, leverage developer communities, and get self-service help

Name	Description
Support	From basic free help to paid packages