Build a Data Warehouse with BigQuery: Challenge Lab

Task 1

Create a table partitioned by date

The starting point for the machine learning model will be the **oxford_policy_tracker** table in the COVID 19 Government Response public dataset which contains details of different actions taken by governments to curb the spread of Covid-19 in their jurisdictions.

Given the fact that there will be models based on a range of time periods, you have to create a dataset and then create a date partitioned version of the **oxford_policy_tracker** table in your newly created dataset, with an expiry time set to **1445** days.

While creating a table, you have also been instructed to exclude the United Kingdom (alpha_3_code=**GBR**), Brazil (alpha_3_code=**BRA**), Canada (alpha_3_code=**CAN**) & the United States of America (alpha_3_code=**USA**) as these will be subject to more in-depth analysis through nation and state specific analysis.

1. Create a new dataset **covid** and create a table **oxford_policy_tracker** in that dataset partitioned by date, with an expiry of **1445** days. The table should initially use the schema defined for the **oxford_policy_tracker** table in the COVID 19 Government Response public dataset .
2. You must also populate the table with the data from the source table for all countries and exclude the United Kingdom (**GBR**), Brazil (**BRA**), Canada (**CAN**) and the United States (**USA**) as instructed above.

Open the BigQuery console

1. In the Google Cloud Console, select **Navigation menu** > **BigQuery**.

The **Welcome to BigQuery in the Cloud Console** message box opens. This message box provides a link to the quickstart guide and the release notes.

2. Click **Done**.

The BigQuery console opens.

Task 1. Create a new dataset

1. First, you will create a dataset to store your tables.

2. In the **Explorer pane**, near your project id, click on **View actions** then click **Create dataset**.

3. Set **Dataset ID** to **covid**.

Leave the other options at their default values (Data Location, Default table Expiration).

4. Click **Create dataset**.

```
CREATE OR REPLACE TABLE <dataset_name>.<table_name>

PARTITION BY date

OPTIONS(

partition_expiration_days=360,

description="oxford_policy_tracker table in the COVID 19 Government Response public dataset with  an expiry time set to 90 days."

) AS

SELECT

   *

FROM

   `bigquery-public-data.covid19_govt_response.oxford_policy_tracker`

WHERE

   alpha_3_code NOT IN ('GBR', 'BRA', 'CAN','USA')
```

Task 2

Populate the mobility record data

In this task, you need to add the **mobility** record data, which requires to extract average values for the six component fields that comprise the mobility record data from the **mobility_report** table from the [Google COVID 19 Mobility public dataset].

Your coworker has also given you a SQL snippet that is currently being used to analyze trends in the Google Mobility data daily mobility patterns. You might need to use this as part of the query that will add the daily country data for the mobility record in table provided in the task description.

```
SELECT country_region, date,
AVG(retail_and_recreation_percent_change_from_baseline) as avg_retail,
AVG(grocery_and_pharmacy_percent_change_from_baseline) as avg_grocery,
AVG(parks_percent_change_from_baseline) as avg_parks,
AVG(transit_stations_percent_change_from_baseline) as avg_transit,
AVG( workplaces_percent_change_from_baseline ) as avg_workplace,
AVG( residential_percent_change_from_baseline) as avg_residential
FROM `bigquery-public-data.covid19_google_mobility.mobility_report`
GROUP BY country_region, date
```

1. Verify the pre-created BigQuery dataset '**covid_data**' within this dataset, populate the mobility record in '**consolidate_covid_tracker_data**' table with data from the [Google COVID 19 Mobility public dataset].

Note: In case you're unable to view pre-created resources in bigquery as per the task description,"your Google Cloud resources are still being provisioned, please refresh the page and try again in a few minutes." If you do, just wait a short time and reload your page.

```
ALTER TABLE <dataset_name>.<table_name>

ADD COLUMN population INT64,

ADD COLUMN country_area FLOAT64,

ADD COLUMN mobility STRUCT<

  avg_retail      FLOAT64,

  avg_grocery     FLOAT64,

  avg_parks       FLOAT64,

  avg_transit     FLOAT64,

  avg_workplace   FLOAT64,
   avg_residential FLOAT64
```

```
CREATE OR REPLACE TABLE <dataset_name>.pop_data_2019 AS
SELECT
  country_territory_code,
  pop_data_2019
FROM
  `bigquery-public-data.covid19_ecdc.covid_19_geographic_distribution_worldwide`
GROUP BY
  country_territory_code,
  pop_data_2019
ORDER BY
  country_territory_code
```

2nd query

```
UPDATE
   `<dataset_name>.<table_name>` t0
SET
   population = t1.pop_data_2019
FROM
   `<dataset_name>.pop_data_2019` t1
WHERE
   CONCAT(t0.alpha_3_code) = CONCAT(t1.country_territory_code);
```

Task 3

Query missing data in population & country_area columns

In this task, you need to find out the countries which do not have population data and countries that do not have country area information.

1. Within the BigQuery dataset named '**covid_data**' contains one table named **oxford_policy_tracker_worldwide**, run a query to find the missing countries in the population and country_area data from '**oxford_policy_tracker_worldwide**' table . The query should list countries that do not have any population data and countries that do not have country area information, ordered by country name. If a country has neither population or country area it must appear twice.

> **Note:** In case you're unable to view pre-created resources in bigquery as per the task description,"your Google Cloud resources are still being provisioned, please refresh the page and try again in a few minutes." If you do, just wait a short time and reload your page.

```
UPDATE
  `<dataset_name>.<table_name>` t0
SET
  t0.country_area = t1.country_area
FROM
  `bigquery-public-data.census_bureau_international.country_names_area` t1
WHERE
  t0.country_name = t1.country_name
```

```
UPDATE
  `<dataset_name>.<table_name>` t0
```

```
SET
  t0.mobility.avg_retail      = t1.avg_retail,
  t0.mobility.avg_grocery     = t1.avg_grocery,
  t0.mobility.avg_parks       = t1.avg_parks,
  t0.mobility.avg_transit     = t1.avg_transit,
  t0.mobility.avg_workplace   = t1.avg_workplace,
  t0.mobility.avg_residential = t1.avg_residential
FROM
  ( SELECT country_region, date,
    AVG(retail_and_recreation_percent_change_from_baseline) as avg_retail,
    AVG(grocery_and_pharmacy_percent_change_from_baseline)  as avg_grocery,
    AVG(parks_percent_change_from_baseline) as avg_parks,
    AVG(transit_stations_percent_change_from_baseline) as avg_transit,
    AVG(workplaces_percent_change_from_baseline) as avg_workplace,
    AVG(residential_percent_change_from_baseline)  as avg_residential
    FROM `bigquery-public-data.covid19_google_mobility.mobility_report`
    GROUP BY country_region, date
  ) AS t1
WHERE
  CONCAT(t0.country_name, t0.date) = CONCAT(t1.country_region, t1.date)
```

Task 4

Create a new table for country population data

In this step, you need to create a copy of **covid_19_geographic_distribution_worldwide** table from **European Center for Disease Control COVID 19 public dataset** into your dataset provided in the task description.

1. Create a new table '**pop_data_2019**' within the dataset named as '**covid_data**'. The table should initially use the schema defined for the '**covid_19_geographic_distribution_worldwide**' table data from the European Center for Disease Control COVID 19 public dataset.
2. Add the country population data to the '**pop_data_2019**' table with **covid_19_geographic_distribution_worldwide** table data from the European Center for Disease Control COVID 19 public dataset.

Note: In case you're unable to view pre-created resources in bigquery as per the task description,"your Google Cloud resources are still being provisioned, please refresh the page and try again in a few minutes." If you do, just wait a short time and reload your page.

```
SELECT country_name, population

FROM `<dataset_name>.<table_name>`

WHERE population is NULL


QUERY 2 ----------------------------


SELECT country_name, country_area

FROM `<dataset_name>.<table_name>`

WHERE country_area IS NULL


QUERY 3 -----------------------------


SELECT DISTINCT country_name

FROM `<dataset_name>.<table_name>`
```

```
WHERE population is NULL
UNION ALL
SELECT DISTINCT country_name
FROM `<dataset_name>.<table_name>`
WHERE country_area IS NULL
ORDER BY country_name ASC
```