```
ON
  midyearpop.country_name = mortality.country_name
GROUP BY
  country,
  year
ORDER BY
  population DESC
LIMIT
  20'

Waiting on bqjob_r5d381c26fcb6480e_0000016628e220c3_1 ... (0s) Current
status: DONE
+---------+------+-------------+--------------------+
| country | year |  population  |  life_expectancy   |
+---------+------+-------------+--------------------+
| Nigeria | 2018 | 2.03452505E8 | 53.483061224489774 |
| Egypt   | 2018 |  9.9413317E7 |   73.8963636363636 |
+---------+------+-------------+--------------------+
```

The Case Against Running Select *

In BigQuery, it is ill-advised to run the SELECT * command, which is used in SQL to retrieve all the columns from the table. This command is rather expensive in BigQuery especially if your table contains terabytes of data. If instead you want to have a feel for the columns and their entries in your dataset, you can execute the command ‘bq head [table_name]’ to retrieve the first few rows of the table. As an example, we used the command in the following example listing to retrieve the first few rows of the ‘market’ table we earlier loaded from GCS in the ‘crypto_data’ dataset.

```
bq head crypto_data.markets
```

```
+------+--------+------+------------+---------+----------+----------+----------+----------+----------+
-----------+-------------+--------+
| slug | symbol | name |    date    | ranknow |   open   |   high   |   low    |  close   |  volume  |
  market   | close_ratio | spread |
+------+--------+------+------------+---------+----------+----------+----------+----------+----------+
-----------+-------------+--------+
| 0x   | ZRX    | 0x   | 2017-08-16 | 41      | 0.111725 | 0.280031 | 0.103962 | 0.224399 | 5232600  |
 67034800  | 0.684       | 0.18   |
| 0x   | ZRX    | 0x   | 2017-08-17 | 41      | 0.223022 | 0.238935 | 0.206735 | 0.206735 | 2752410  |
 133813000 | 0           | 0.03   |
| 0x   | ZRX    | 0x   | 2017-08-18 | 41      | 0.205558 | 0.35026  | 0.205558 | 0.293387 | 12793800 |
 123335000 | 0.607       | 0.14   |
......
......
| 0x   | ZRX    | 0x   | 2017-08-28 | 41      | 0.352459 | 0.354823 | 0.32362  | 0.343713 | 6639910  |
 176230000 | 0.6439      | 0.03   |
+------+--------+------+------------+---------+----------+----------+----------+----------+----------+
-----------+-------------+--------+
```

Using BigQuery with Notebooks on AI Cloud Instance and Google Colab

BigQuery integrates well with Notebooks on Google Notebook AI Instance and Google Colab. In this section, we'll go through executing on BigQuery datasets and tables from Notebooks. There are a couple of ways to interact with BigQuery from Notebooks, but one quick and easy method is the use of the **'%bigquery'** magic command from the BigQuery client library, **'google-cloud-bigquery'**, to run queries with minimal syntax.

The **%%bigquery** magic runs a SQL query and returns the results as a pandas DataFrame. Here, we use the '%%bigquery' magic command to interact with BigQuery. To begin, open a Notebook on GCP AI Notebook Instance or from Colab:

1. If running on Google Colab, authenticate the notebook by running the code

   from google.colab import auth
   auth.authenticate_user()
   print('Authenticated')

2. Import Pandas and Matplotlib.

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

3. Store the following query output as a Pandas DataFrame named **'litcoin_crypto'**. Place your project id after the **'--project'** attribute. Be sure to update the FROM field with your dataset and table IDs.

   ```
   %%bigquery --project ekabasandbox litcoin_crypto
   SELECT
     symbol,
     date,
     close,
     open,
     high,
     low,
     spread
   ```