



MENTOR NESS

# **CORONA VIRUS ANALYSIS WITH SQL**

**EXECUTED BY**

**Data Analyst Intern  
Fielami Emmanuel David**

**Batch Name:  
MIP-DA-07**

# PROJECT OVERVIEW

- The COVID-19 pandemic has profoundly affected public health, necessitating data-driven insights to comprehend its spread.
- As a data analyst, the task is to analyze a COVID-19 dataset using SQL to derive meaningful insights and present findings effectively.



# OBJECTIVE

The objective is to gain a deeper understanding of the COVID-19 pandemic's impact through SQL data analysis. Key metrics such as confirmed cases, deaths, and recoveries will be examined to identify trends, patterns, and correlations in the spread of the virus.

# DATASET DESCRIPTION

Description of each column in the dataset (Corona Virus Dataset)

- Province: Geographic subdivision within a country/region.
- Country/Region: Geographic entity where data is recorded.
- Latitude: North-south position on Earth's surface.
- Longitude: East-west position on Earth's surface.
- Date: Recorded date of CORONA VIRUS data.
- Confirmed: Number of diagnosed CORONA VIRUS cases.
- Deaths: Number of CORONA VIRUS-related deaths.
- Recovered: Number of recovered CORONA VIRUS cases



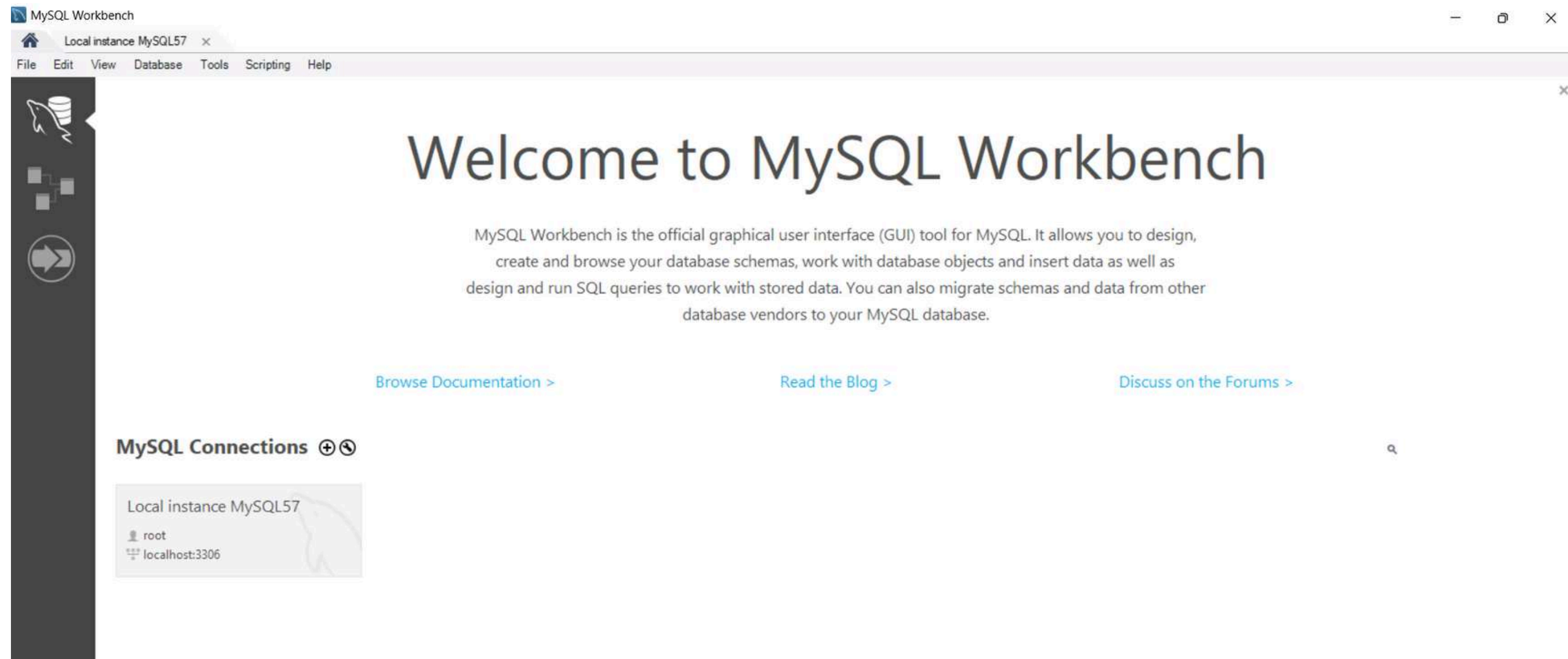
# **DATABASE SETUP AND TOOL UTILIZED**

**Let's get into it!**



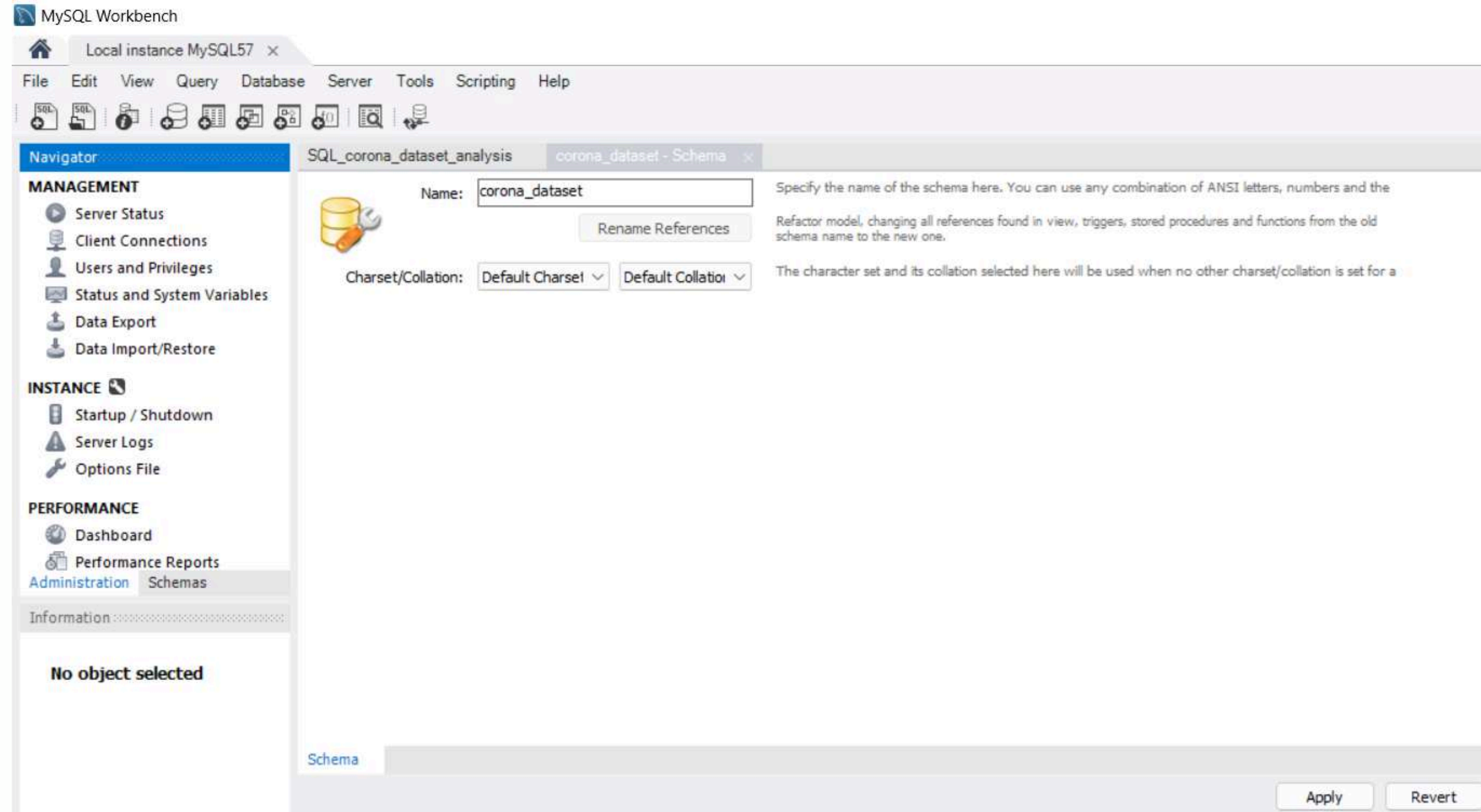
# DATABASE SETUP

The database management system utilized is MySQL Workbench.



# CREATING DATABASE

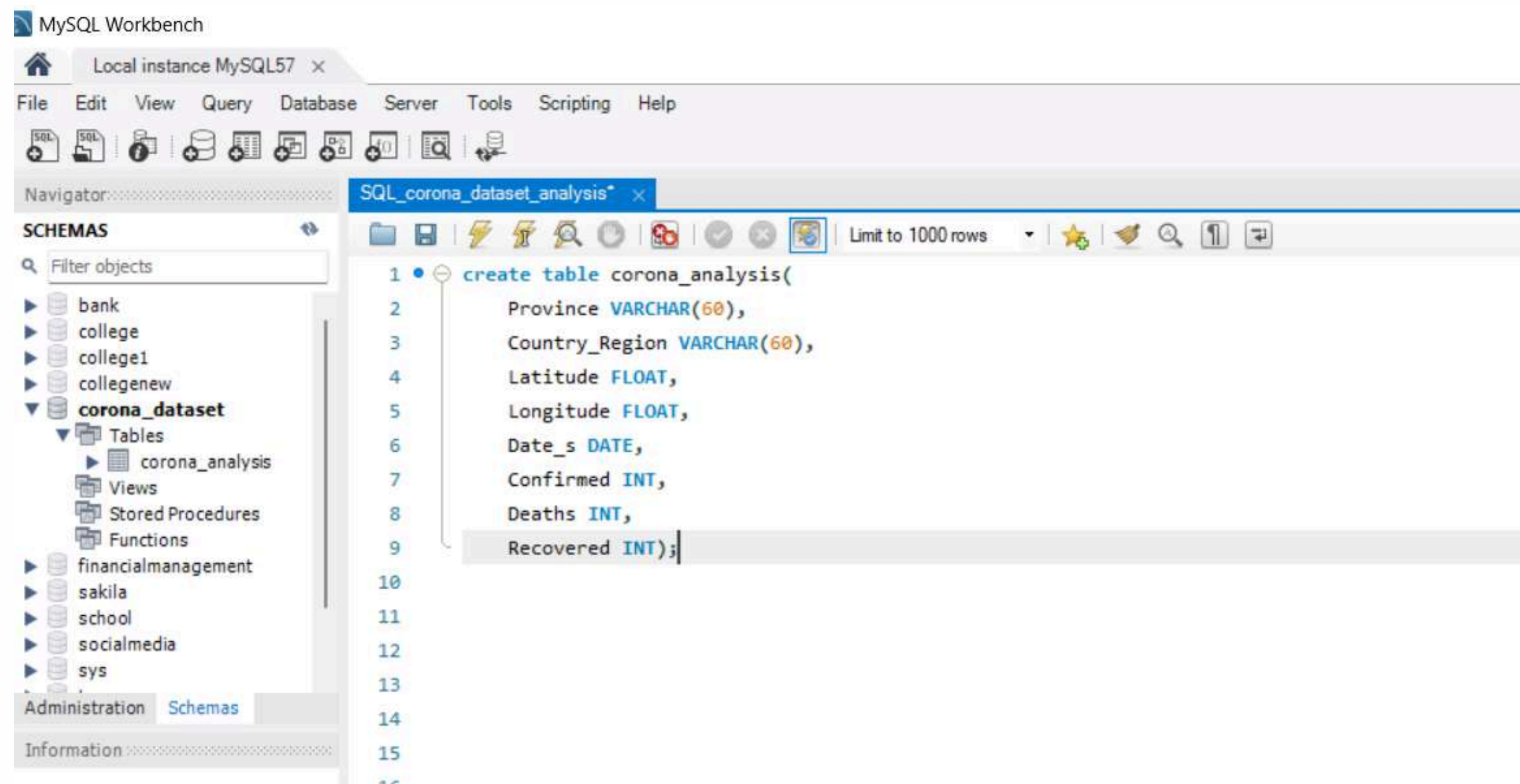
The database name is "corona\_dataset".





# CREATING TABLE AND DATA IMPORT

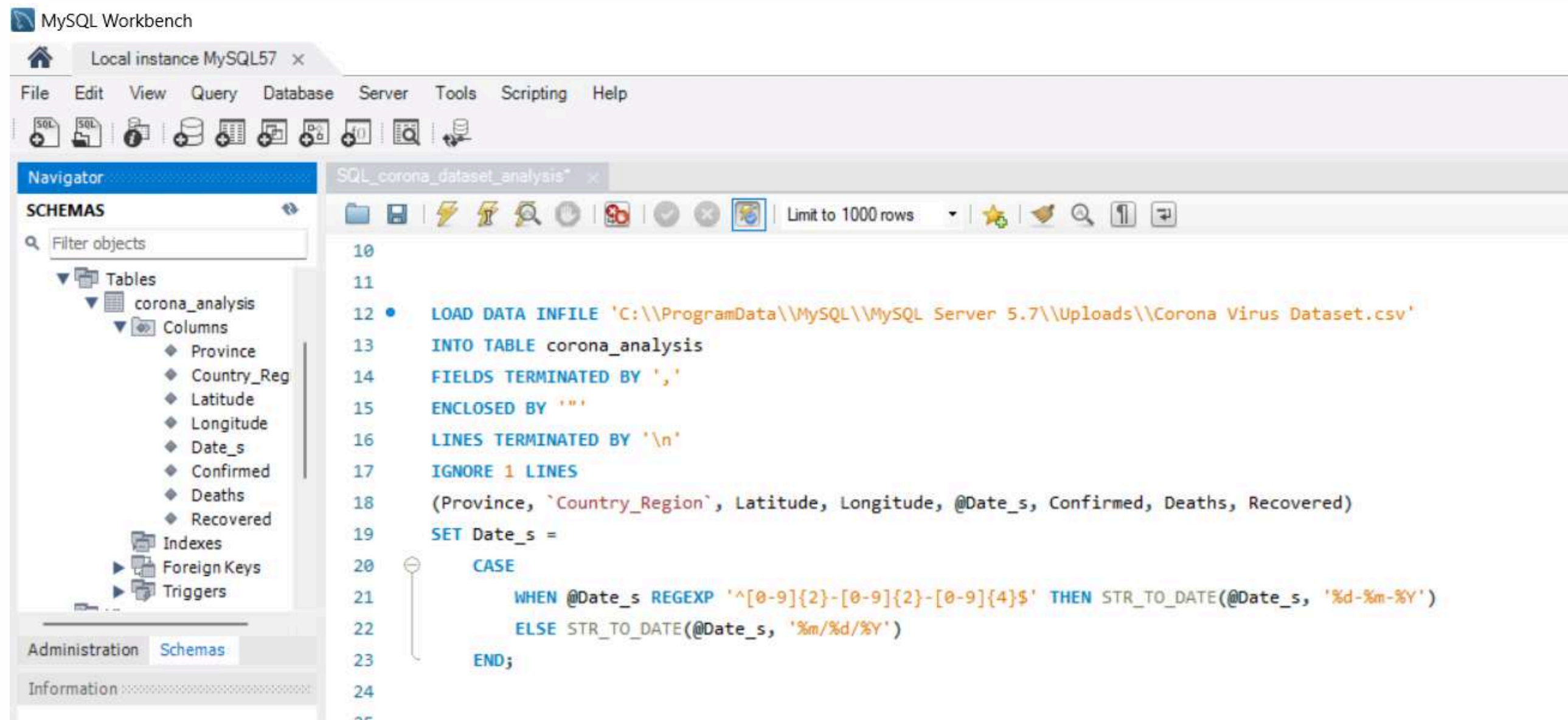
The provided code is a SQL query used to create a table named "corona\_analysis"





# CREATING TABLE AND DATA IMPORT

The provided code is a SQL query used to load data from a CSV file into the "corona\_analysis" table.

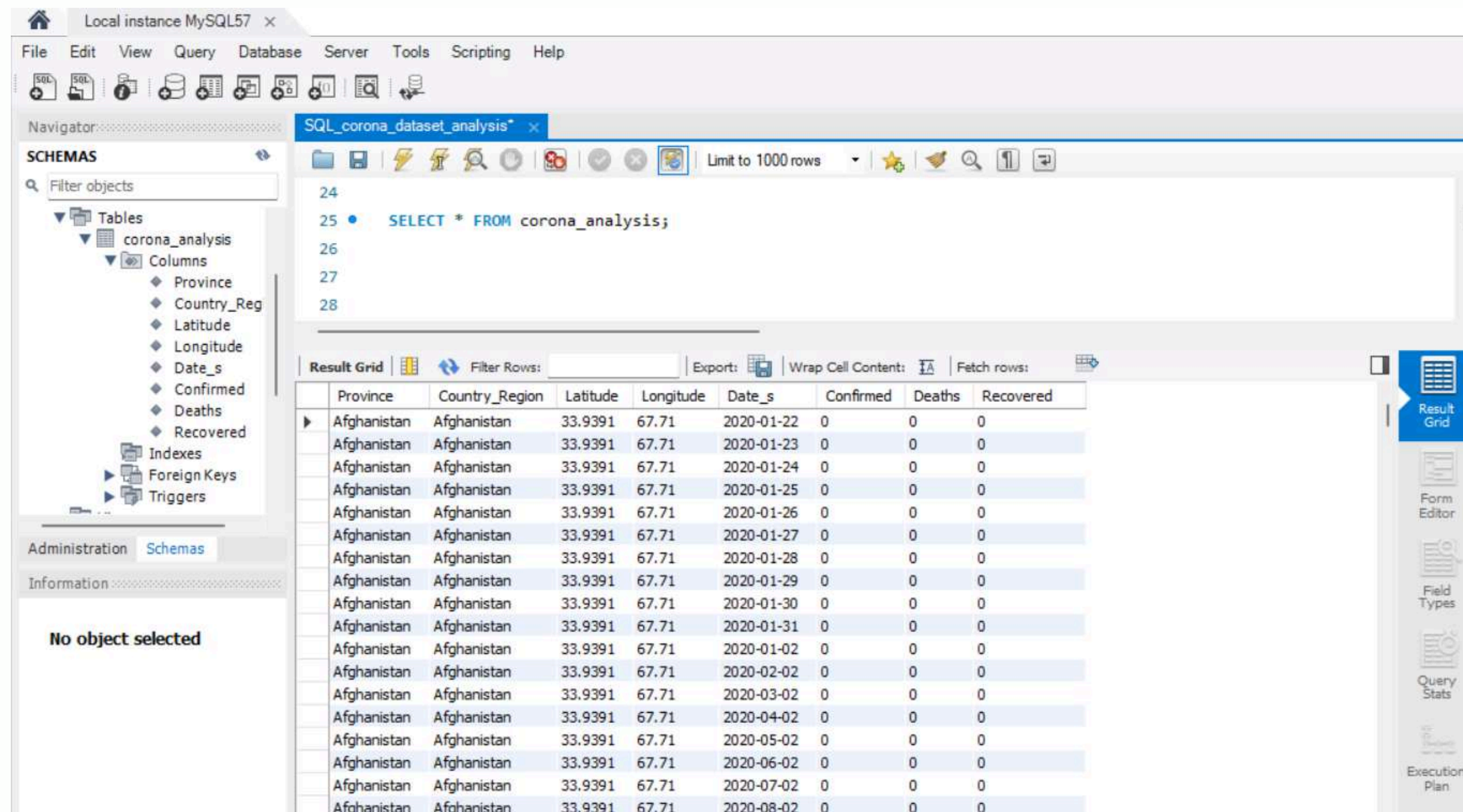


The screenshot displays the MySQL Workbench interface. On the left, the 'Navigator' pane shows the 'SCHEMAS' tree with 'Tables' expanded, revealing the 'corona\_analysis' table and its columns: Province, Country\_Reg, Latitude, Longitude, Date\_s, Confirmed, Deaths, and Recovered. The main editor window, titled 'SQL\_corona\_dataset\_analysis', contains the following SQL query:

```
10
11
12 • LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 5.7\\Uploads\\Corona Virus Dataset.csv'
13 INTO TABLE corona_analysis
14 FIELDS TERMINATED BY ','
15 ENCLOSED BY '"'
16 LINES TERMINATED BY '\n'
17 IGNORE 1 LINES
18 (Province, `Country_Region`, Latitude, Longitude, @Date_s, Confirmed, Deaths, Recovered)
19 SET Date_s =
20 CASE
21     WHEN @Date_s REGEXP '^[0-9]{2}-[0-9]{2}-[0-9]{4}$' THEN STR_TO_DATE(@Date_s, '%d-%m-%Y')
22     ELSE STR_TO_DATE(@Date_s, '%m/%d/%Y')
23 END;
24
```

# DISPLAY THE IMPORTED DATA

The provided SQL code is a query used to retrieve all records from the "corona\_analysis" table



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'corona\_analysis' table expanded, showing columns: Province, Country\_Reg, Latitude, Longitude, Date\_s, Confirmed, Deaths, and Recovered. The main window shows a SQL query: `SELECT * FROM corona_analysis;`. Below the query, the 'Result Grid' displays 15 rows of data for Afghanistan, all with zero confirmed cases, deaths, and recoveries.

Province	Country_Region	Latitude	Longitude	Date_s	Confirmed	Deaths	Recovered
Afghanistan	Afghanistan	33.9391	67.71	2020-01-22	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-23	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-24	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-25	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-26	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-27	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-28	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-29	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-30	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-31	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-01-02	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-02-02	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-03-02	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-04-02	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-05-02	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-06-02	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-07-02	0	0	0
Afghanistan	Afghanistan	33.9391	67.71	2020-08-02	0	0	0

# DATA ANALYSIS

Let's get into it!

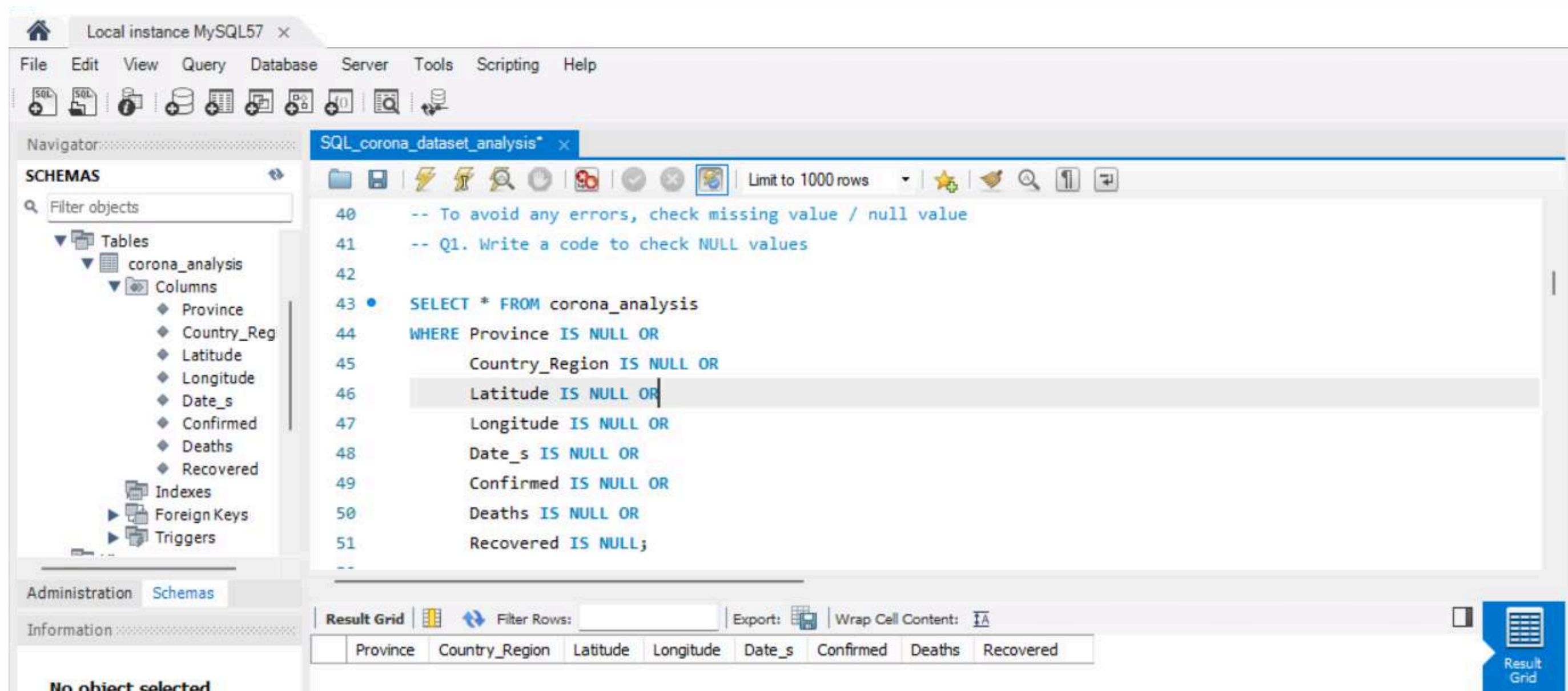




# DATA CLEANING

To prevent potential errors in the future, let's ensure there are no missing or null values in the dataset.

Q1. Write a code to check NULL values





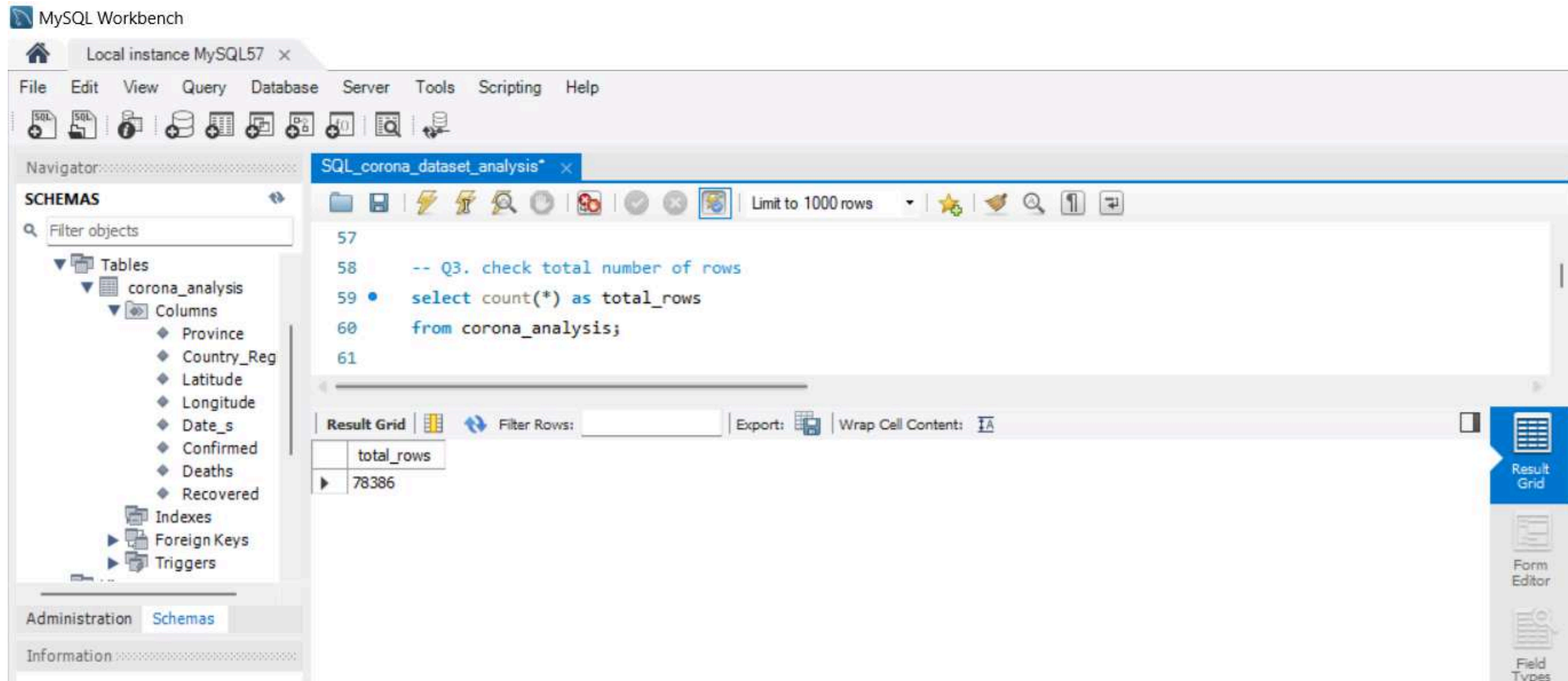


Q2. IF NULL VALUES ARE PRESENT, UPDATE THEM WITH ZEROS  
FOR ALL COLUMNS

Based on the Output: Since there are no NULL values present in  
the dataset, we don't need to perform any updates.



### Q3. CHECK TOTAL NUMBER OF ROWS.



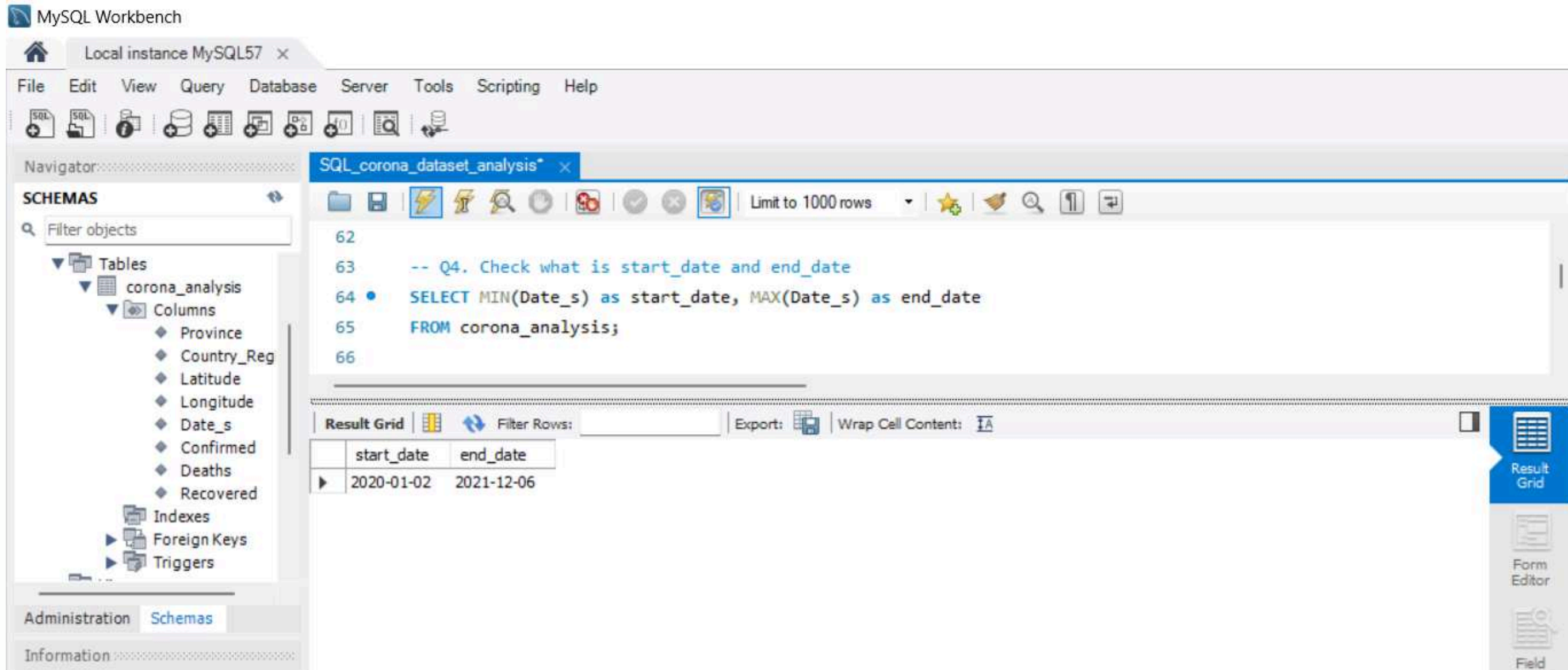
The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays the 'corona\_analysis' database with columns: Province, Country\_Reg, Latitude, Longitude, Date\_s, Confirmed, Deaths, and Recovered. The main editor window, titled 'SQL\_corona\_dataset\_analysis\*', contains the following SQL query:

```
57  
58  -- Q3. check total number of rows  
59  • select count(*) as total_rows  
60  from corona_analysis;  
61
```

Below the query editor, the 'Result Grid' is visible, showing a single row with the column 'total\_rows' and the value '78386'. The interface also includes a menu bar (File, Edit, View, Query, Database, Server, Tools, Scripting, Help) and a toolbar with various icons for file operations, query execution, and result viewing.

Insight: The dataset contains a total of 78,386 records.

## Q4. CHECK WHAT IS START\_DATE AND END\_DATE



The screenshot shows the MySQL Workbench interface. On the left, the Navigator pane displays the 'corona\_analysis' table with columns: Province, Country\_Reg, Latitude, Longitude, Date\_s, Confirmed, Deaths, and Recovered. The main editor window shows a SQL query:

```
-- Q4. Check what is start_date and end_date
SELECT MIN(Date_s) as start_date, MAX(Date_s) as end_date
FROM corona_analysis;
```

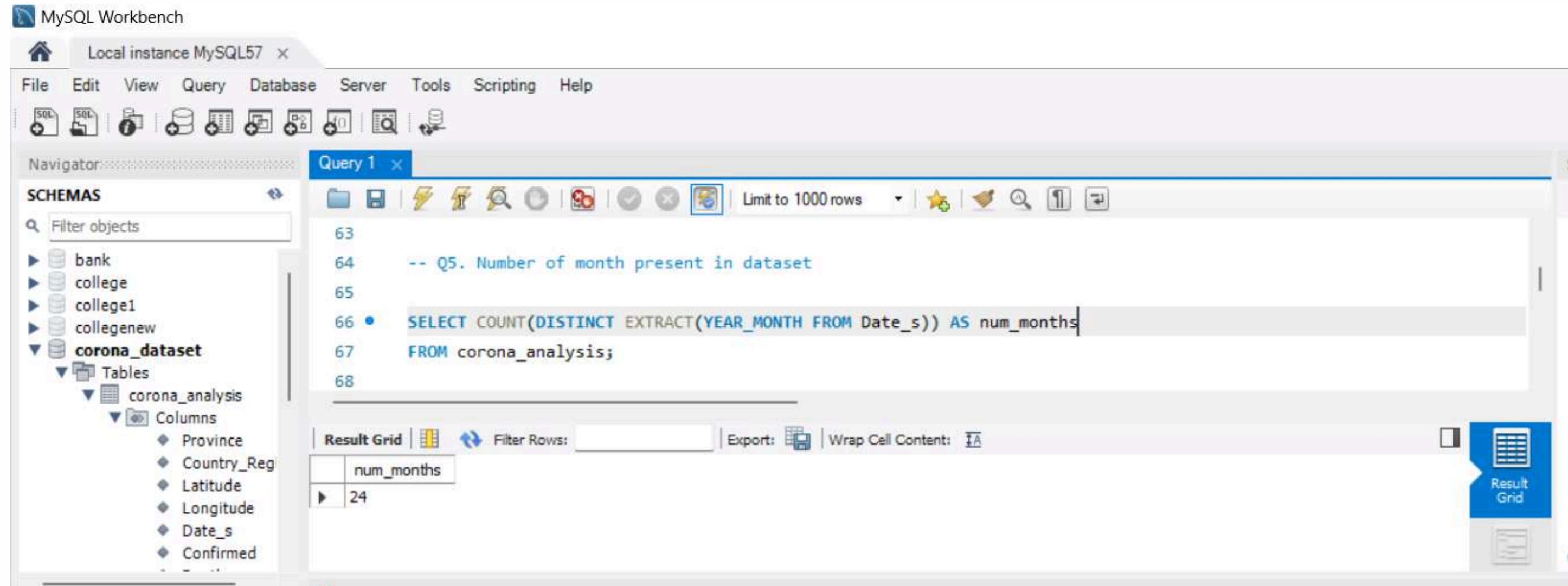
The Result Grid at the bottom shows the output of the query:

start_date	end_date
2020-01-02	2021-12-06

Insight: Based on the Dataset the start date of the COVID-19 is recorded as January 02, 2020 with the end date as June 12, 2021



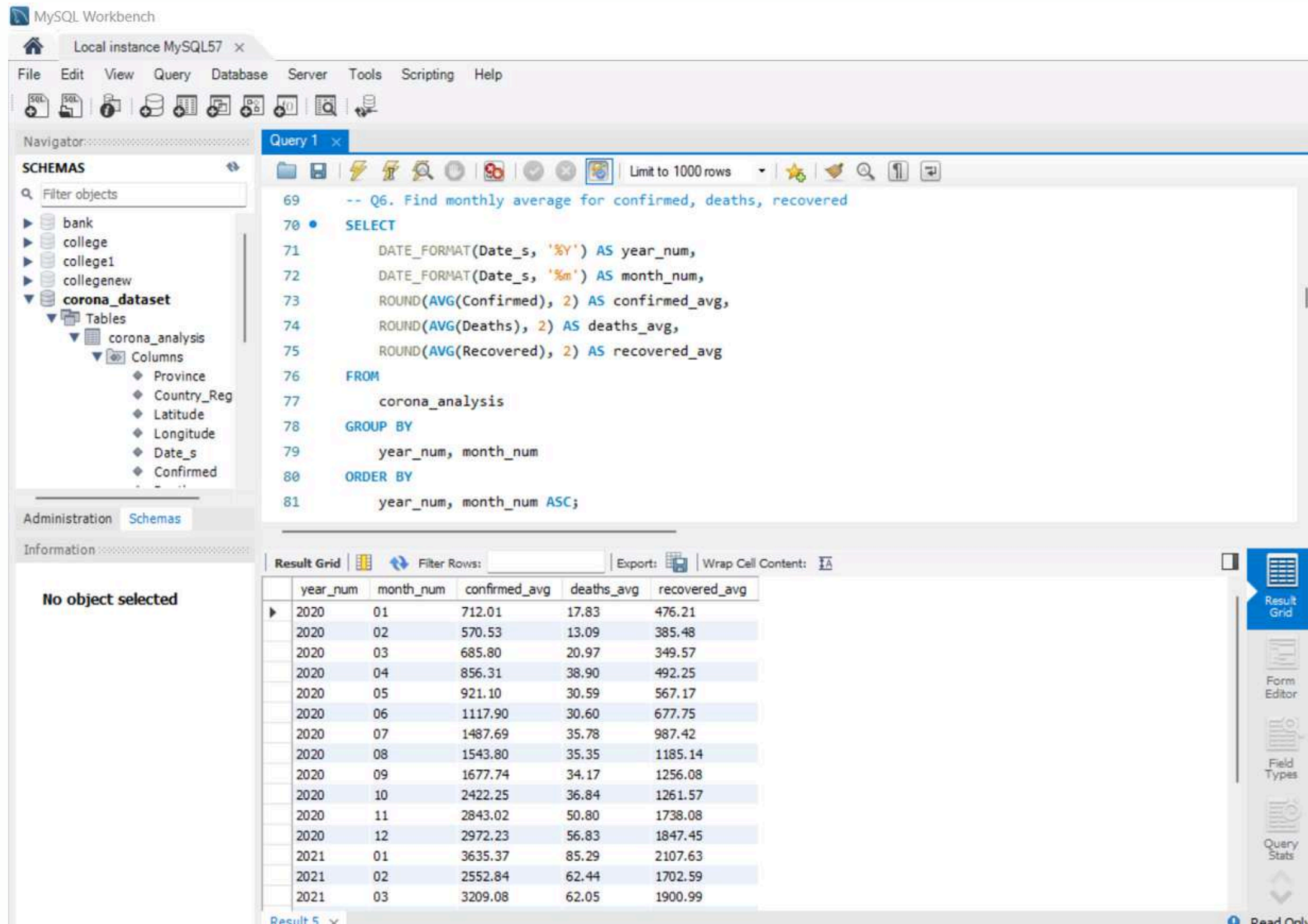
## Q5. NUMBER OF MONTHS PRESENT IN THE DATASE



Insight: The output of this query provides the total number of unique months present in the dataset.



## Q6. FIND THE MONTHLY AVERAGE FOR CONFIRMED, DEATHS, RECOVERED



The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a tree view with 'corona\_dataset' expanded, showing 'corona\_analysis' and its columns: 'Province', 'Country\_Reg', 'Latitude', 'Longitude', 'Date\_s', and 'Confirmed'. The 'Query 1' editor shows the following SQL query:

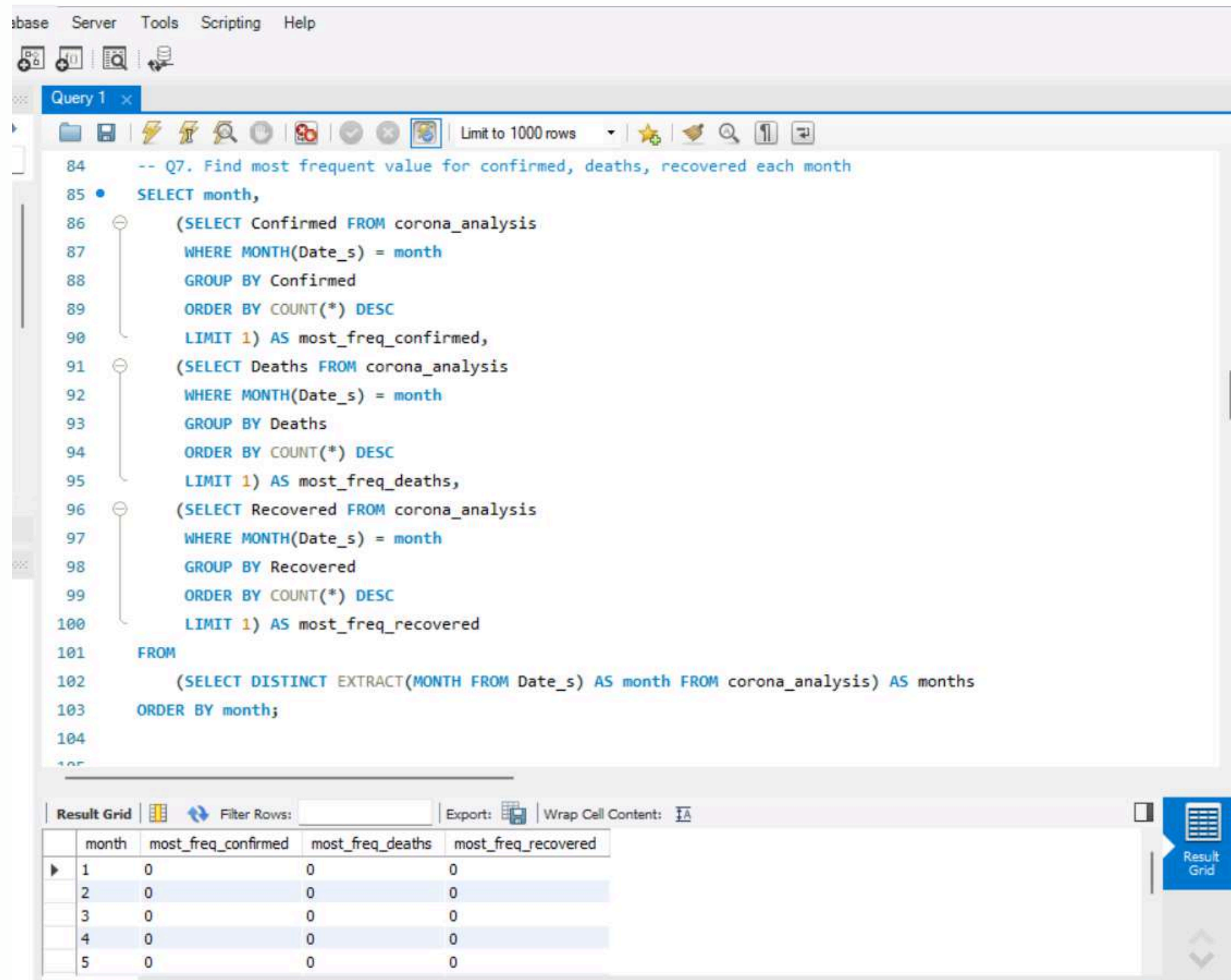
```
-- Q6. Find monthly average for confirmed, deaths, recovered
SELECT
  DATE_FORMAT(Date_s, '%Y') AS year_num,
  DATE_FORMAT(Date_s, '%m') AS month_num,
  ROUND(AVG(Confirmed), 2) AS confirmed_avg,
  ROUND(AVG(Deaths), 2) AS deaths_avg,
  ROUND(AVG(Recovered), 2) AS recovered_avg
FROM
  corona_analysis
GROUP BY
  year_num, month_num
ORDER BY
  year_num, month_num ASC;
```

The 'Result Grid' at the bottom displays the results of the query, showing the monthly average for confirmed cases, deaths, and recoveries for each month from 2020 to 2021.

year_num	month_num	confirmed_avg	deaths_avg	recovered_avg
2020	01	712.01	17.83	476.21
2020	02	570.53	13.09	385.48
2020	03	685.80	20.97	349.57
2020	04	856.31	38.90	492.25
2020	05	921.10	30.59	567.17
2020	06	1117.90	30.60	677.75
2020	07	1487.69	35.78	987.42
2020	08	1543.80	35.35	1185.14
2020	09	1677.74	34.17	1256.08
2020	10	2422.25	36.84	1261.57
2020	11	2843.02	50.80	1738.08
2020	12	2972.23	56.83	1847.45
2021	01	3635.37	85.29	2107.63
2021	02	2552.84	62.44	1702.59
2021	03	3209.08	62.05	1900.99

This query extracts the year and month from the Date\_s column, calculates the average number of confirmed cases (Confirmed), deaths (Deaths), and recoveries (Recovered) for each month, and rounds the averages to two decimal places

## Q7. FIND MOST FREQUENT VALUE FOR CONFIRMED, DEATHS, RECOVERED EACH MONTH.



The screenshot shows a database query editor with a menu bar (Database, Server, Tools, Scripting, Help) and a toolbar. The query editor displays a SQL query for 'Query 1'. The query is as follows:

```
-- Q7. Find most frequent value for confirmed, deaths, recovered each month
SELECT month,
  (SELECT Confirmed FROM corona_analysis
   WHERE MONTH(Date_s) = month
   GROUP BY Confirmed
   ORDER BY COUNT(*) DESC
   LIMIT 1) AS most_freq_confirmed,
  (SELECT Deaths FROM corona_analysis
   WHERE MONTH(Date_s) = month
   GROUP BY Deaths
   ORDER BY COUNT(*) DESC
   LIMIT 1) AS most_freq_deaths,
  (SELECT Recovered FROM corona_analysis
   WHERE MONTH(Date_s) = month
   GROUP BY Recovered
   ORDER BY COUNT(*) DESC
   LIMIT 1) AS most_freq_recovered
FROM
  (SELECT DISTINCT EXTRACT(MONTH FROM Date_s) AS month FROM corona_analysis) AS months
ORDER BY month;
```

The results are displayed in a table with the following columns: month, most\_freq\_confirmed, most\_freq\_deaths, and most\_freq\_recovered. The table shows 5 rows of data, all with a value of 0 for the frequency columns.

month	most_freq_confirmed	most_freq_deaths	most_freq_recovered
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0

The query extracts unique months, labels them, finds the most frequent values for confirmed cases, deaths, and recoveries, and sorts by month.



## Q8. FIND MINIMUM VALUES FOR CONFIRMED, DEATHS, RECOVERED PER YEAR.

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays a tree view of databases. The 'corona\_dataset' database is selected, and its 'Tables' folder is expanded, showing the 'corona\_analysis' table. The 'Columns' folder for 'corona\_analysis' is also expanded, listing 'Province', 'Country\_Reg', 'Latitude', 'Longitude', 'Date\_s', and 'Confirmed'. The main query editor displays the following SQL query:

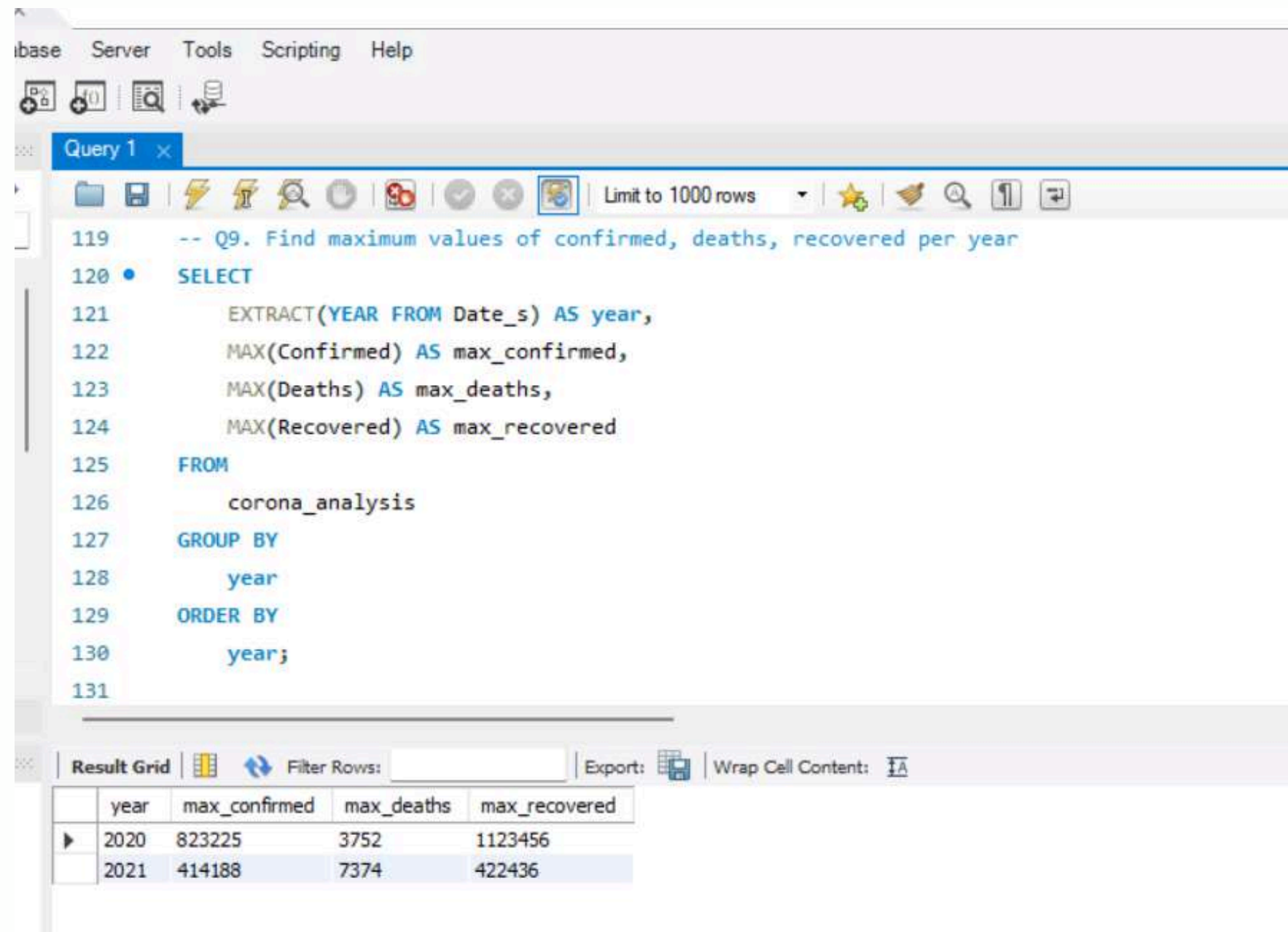
```
-- Q8. Find minimum values for confirmed, deaths, recovered per year
SELECT
    EXTRACT(YEAR FROM Date_s) AS year,
    MIN(Confirmed) AS min_confirmed,
    MIN(Deaths) AS min_deaths,
    MIN(Recovered) AS min_recovered
FROM
    corona_analysis
GROUP BY
    year
ORDER BY
    year;
```

Below the query editor, the 'Result Grid' tab is active, showing the results of the query. The results are displayed in a table with the following columns: 'year', 'min\_confirmed', 'min\_deaths', and 'min\_recovered'. The data shows that for both 2020 and 2021, the minimum values for confirmed cases, deaths, and recovered cases are all 0.

year	min_confirmed	min_deaths	min_recovered
2020	0	0	0
2021	0	0	0



## Q9. FIND MAXIMUM VALUES OF CONFIRMED, DEATHS, RECOVERED CASES PER YEAR.



The screenshot shows a database query editor with a menu bar (Database, Server, Tools, Scripting, Help) and a toolbar. The query editor displays a SQL query for 'Query 1' with line numbers 119 to 131. The query is as follows:

```
-- Q9. Find maximum values of confirmed, deaths, recovered per year
119
120 • SELECT
121     EXTRACT(YEAR FROM Date_s) AS year,
122     MAX(Confirmed) AS max_confirmed,
123     MAX(Deaths) AS max_deaths,
124     MAX(Recovered) AS max_recovered
125 FROM
126     corona_analysis
127 GROUP BY
128     year
129 ORDER BY
130     year;
131
```

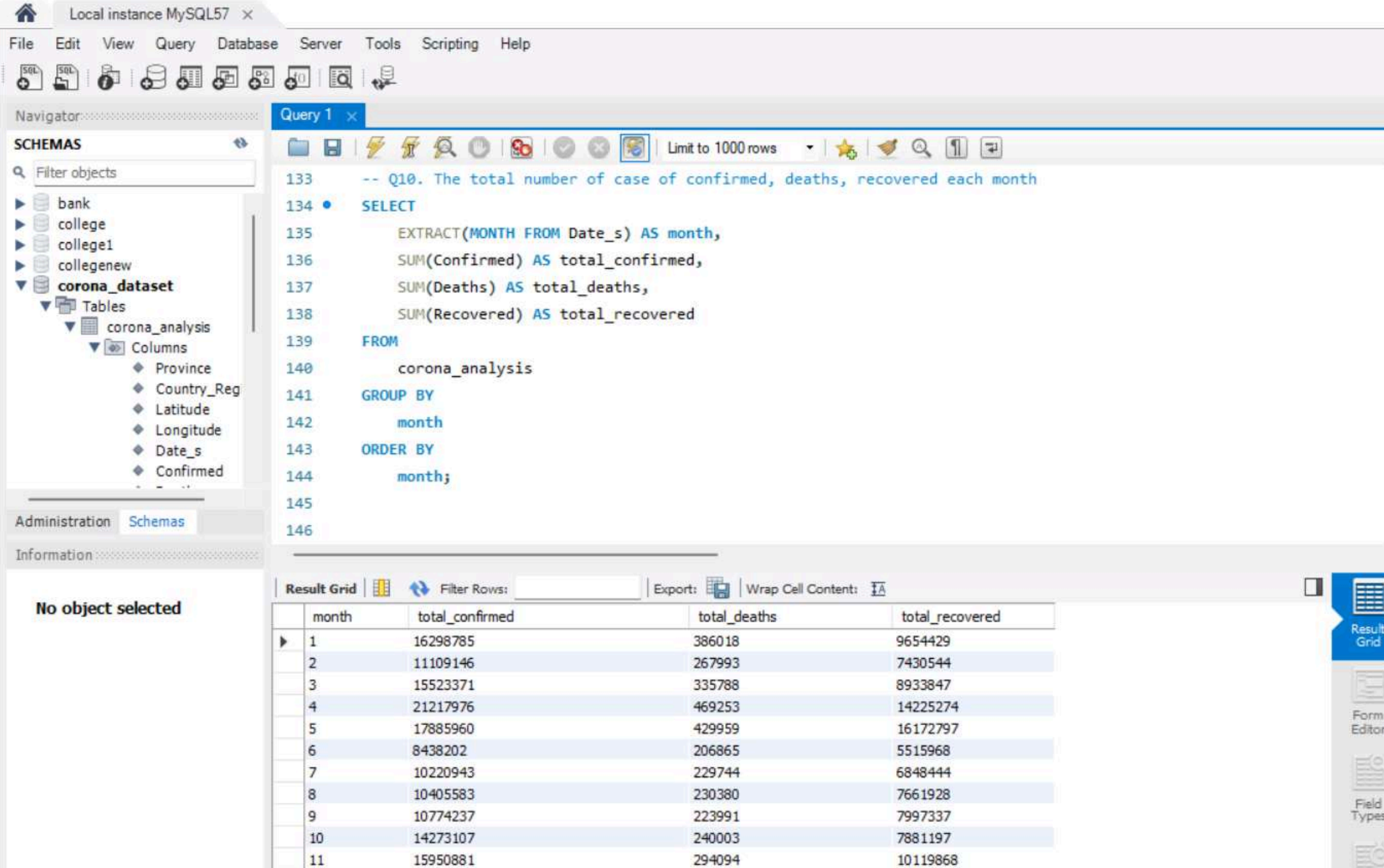
Below the query editor is a 'Result Grid' showing the results of the query. It has columns for 'year', 'max\_confirmed', 'max\_deaths', and 'max\_recovered'. The results are as follows:

year	max_confirmed	max_deaths	max_recovered
2020	823225	3752	1123456
2021	414188	7374	422436

- The year 2020 saw the highest count of confirmed cases, totaling 823,225.
- Conversely, 2021 documented the highest number of deaths at 7,374.
- However, 2020 also recorded the most recoveries, reaching 1,123,456.



# Q10. THE TOTAL NUMBER OF CASE OF CONFIRMED, DEATHS, RECOVERED EACH MONTH.



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with the 'corona\_dataset' database expanded, showing the 'corona\_analysis' table. The main query editor contains the following SQL code:

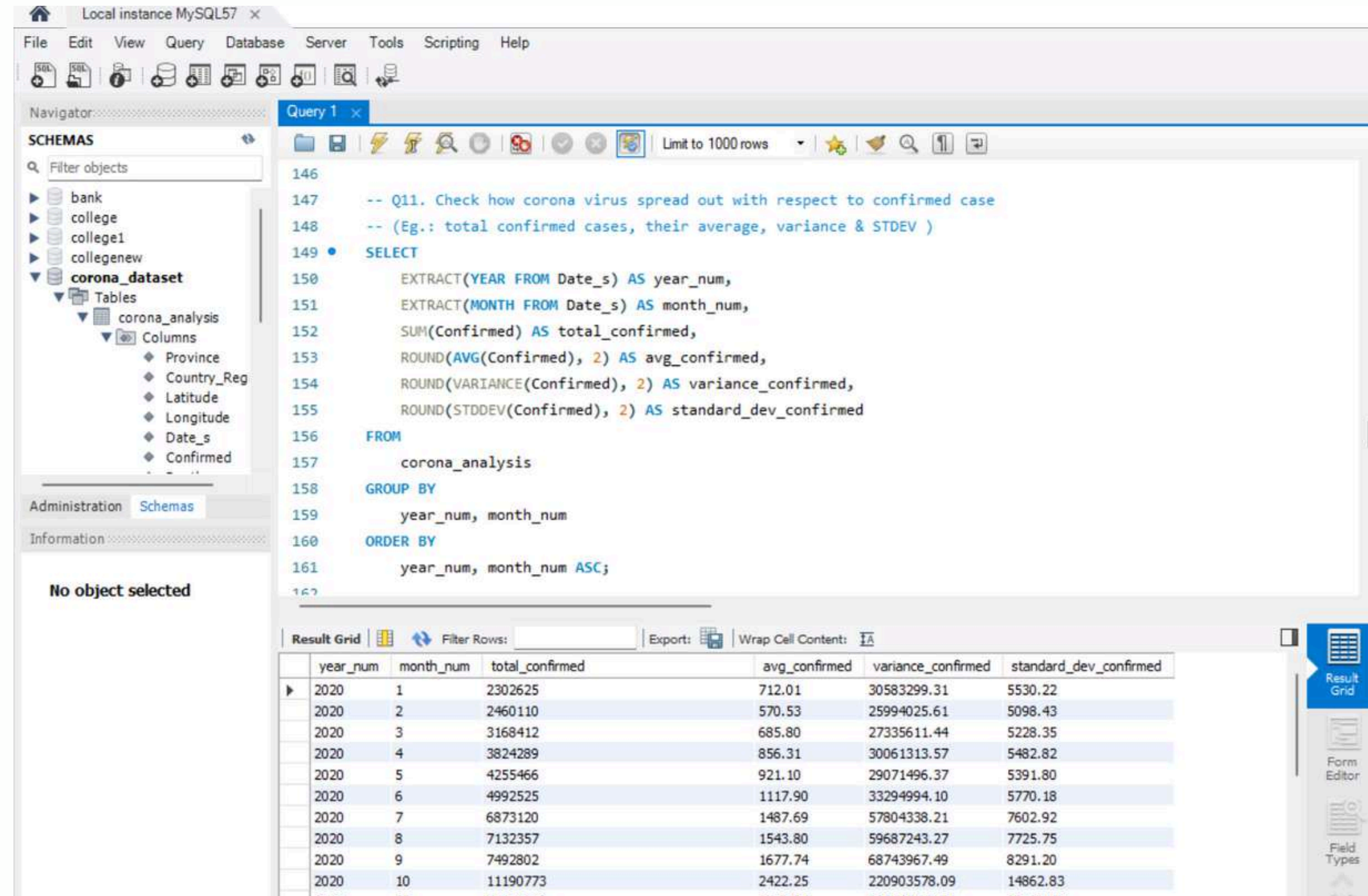
```
-- Q10. The total number of case of confirmed, deaths, recovered each month
SELECT
    EXTRACT(MONTH FROM Date_s) AS month,
    SUM(Confirmed) AS total_confirmed,
    SUM(Deaths) AS total_deaths,
    SUM(Recovered) AS total_recovered
FROM
    corona_analysis
GROUP BY
    month
ORDER BY
    month;
```

The 'Result Grid' at the bottom displays the query results for 11 months. The columns are 'month', 'total\_confirmed', 'total\_deaths', and 'total\_recovered'.

month	total_confirmed	total_deaths	total_recovered
1	16298785	386018	9654429
2	11109146	267993	7430544
3	15523371	335788	8933847
4	21217976	469253	14225274
5	17885960	429959	16172797
6	8438202	206865	5515968
7	10220943	229744	6848444
8	10405583	230380	7661928
9	10774237	223991	7997337
10	14273107	240003	7881197
11	15950881	294094	10119868

# Q11. CHECK HOW CORONA VIRUS SPREAD OUT WITH RESPECT TO CONFIRMED CASES PER MONTH.

(EG.: TOTAL CONFIRMED CASES, THEIR AVERAGE, VARIANCE & STDEV )

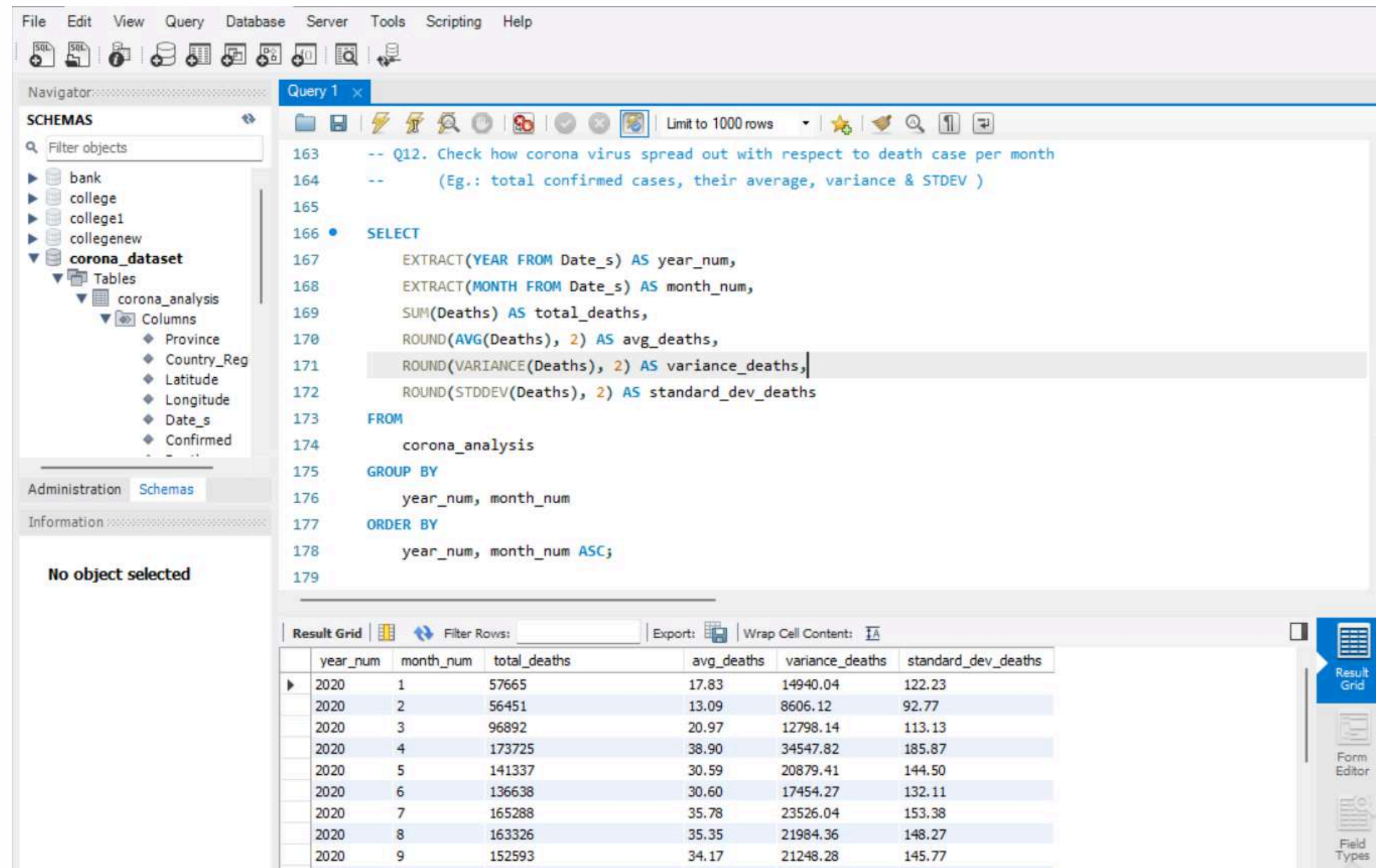


The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays a tree view with 'corona\_dataset' expanded, showing 'corona\_analysis' and its columns: Province, Country\_Reg, Latitude, Longitude, Date\_s, and Confirmed. The main editor shows a SQL query for Q11. The 'Result Grid' at the bottom displays the query results for the first 10 months of 2020.

```
146
147 -- Q11. Check how corona virus spread out with respect to confirmed case
148 -- (Eg.: total confirmed cases, their average, variance & STDEV )
149 • SELECT
150     EXTRACT(YEAR FROM Date_s) AS year_num,
151     EXTRACT(MONTH FROM Date_s) AS month_num,
152     SUM(Confirmed) AS total_confirmed,
153     ROUND(AVG(Confirmed), 2) AS avg_confirmed,
154     ROUND(VARIANCE(Confirmed), 2) AS variance_confirmed,
155     ROUND(STDDEV(Confirmed), 2) AS standard_dev_confirmed
156 FROM
157     corona_analysis
158 GROUP BY
159     year_num, month_num
160 ORDER BY
161     year_num, month_num ASC;
```

year_num	month_num	total_confirmed	avg_confirmed	variance_confirmed	standard_dev_confirmed
2020	1	2302625	712.01	30583299.31	5530.22
2020	2	2460110	570.53	25994025.61	5098.43
2020	3	3168412	685.80	27335611.44	5228.35
2020	4	3824289	856.31	30061313.57	5482.82
2020	5	4255466	921.10	29071496.37	5391.80
2020	6	4992525	1117.90	33294994.10	5770.18
2020	7	6873120	1487.69	57804338.21	7602.92
2020	8	7132357	1543.80	59687243.27	7725.75
2020	9	7492802	1677.74	68743967.49	8291.20
2020	10	11190773	2422.25	220903578.09	14862.83

## Q12. CHECK HOW CORONA VIRUS SPREAD OUT WITH RESPECT TO DEATH CASES PER MONTH. (EG.: TOTAL DEATH CASES, THEIR AVERAGE, VARIANCE & STDEV )



The screenshot shows a database management tool interface. On the left, a 'Navigator' pane displays a tree structure of schemas, including 'corona\_dataset' and its tables. The main area shows a SQL query for 'Query 1' with line numbers 163 to 179. The query calculates monthly statistics for COVID-19 deaths. Below the query, the 'Result Grid' displays the output as a table with 7 columns: year\_num, month\_num, total\_deaths, avg\_deaths, variance\_deaths, and standard\_dev\_deaths. The results show data for the first 9 months of 2020.

```
163 -- Q12. Check how corona virus spread out with respect to death case per month
164 -- (Eg.: total confirmed cases, their average, variance & STDEV )
165
166 • SELECT
167     EXTRACT(YEAR FROM Date_s) AS year_num,
168     EXTRACT(MONTH FROM Date_s) AS month_num,
169     SUM(Deaths) AS total_deaths,
170     ROUND(AVG(Deaths), 2) AS avg_deaths,
171     ROUND(VARIANCE(Deaths), 2) AS variance_deaths,
172     ROUND(STDDEV(Deaths), 2) AS standard_dev_deaths
173 FROM
174     corona_analysis
175 GROUP BY
176     year_num, month_num
177 ORDER BY
178     year_num, month_num ASC;
179
```

year_num	month_num	total_deaths	avg_deaths	variance_deaths	standard_dev_deaths
2020	1	57665	17.83	14940.04	122.23
2020	2	56451	13.09	8606.12	92.77
2020	3	96892	20.97	12798.14	113.13
2020	4	173725	38.90	34547.82	185.87
2020	5	141337	30.59	20879.41	144.50
2020	6	136638	30.60	17454.27	132.11
2020	7	165288	35.78	23526.04	153.38
2020	8	163326	35.35	21984.36	148.27
2020	9	152593	34.17	21248.28	145.77





## Q13. CHECK HOW CORONA VIRUS SPREAD OUT WITH RESPECT TO RECOVERED CASES PER MONTH. (EG.: TOTAL RECOVERED CASES, THEIR AVERAGE, VARIANCE & STDEV )

The screenshot shows a database query editor with a left-hand sidebar containing a 'Navigator' pane. The 'SCHEMAS' section is expanded, showing a tree structure: 'bank', 'college', 'college1', 'collegenew', and 'corona\_dataset'. Under 'corona\_dataset', there are 'Tables' and 'Columns'. The 'Columns' section is expanded, showing 'Province', 'Country\_Reg', 'Latitude', 'Longitude', 'Date\_s', and 'Confirmed'. The 'Administration' tab is selected, and the 'Schemas' sub-tab is active. The 'Information' pane at the bottom left shows 'No object selected'.

The main query editor displays a SQL query for 'Query 1'. The query is as follows:

```
181 -- Q13. Check how corona virus spread out with respect to recovered case
182 -- (Eg.: total confirmed cases, their average, variance & STDEV )
183 SELECT
184     EXTRACT(YEAR FROM Date_s) AS year_num,
185     EXTRACT(MONTH FROM Date_s) AS month_num,
186     SUM(Recovered) AS total_recovered,
187     ROUND(AVG(Recovered), 2) AS avg_recovered,
188     ROUND(VARIANCE(Recovered), 2) AS variance_recovered,
189     ROUND(STDDEV(Recovered), 2) AS standard_dev_recovered
190 FROM
191     corona_analysis
192 GROUP BY
193     year_num, month_num
194 ORDER BY
195     year_num, month_num ASC;
```

The query results are displayed in a 'Result Grid' at the bottom. The grid has 7 columns: 'year\_num', 'month\_num', 'total\_recovered', 'avg\_recovered', 'variance\_recovered', and 'standard\_dev\_recovered'. The data shows the number of recovered cases per month for the year 2020.

year_num	month_num	total_recovered	avg_recovered	variance_recovered	standard_dev_recovered
2020	1	1540072	476.21	16251139.92	4031.27
2020	2	1662179	385.48	15006039.53	3873.76
2020	3	1615017	349.57	9554684.39	3091.07
2020	4	2198371	492.25	12499339.30	3535.44
2020	5	2620319	567.17	11717706.55	3423.11
2020	6	3026824	677.75	10809189.68	3287.73
2020	7	4561886	987.42	23792039.56	4877.71
2020	8	5475343	1185.14	37245189.96	6102.88





## Q14. FIND COUNTRY HAVING HIGHEST NUMBER OF THE CONFIRMED CASE.

The screenshot shows the MySQL Workbench interface for a local instance of MySQL 5.7. The 'Navigator' pane on the left displays the 'corona\_dataset' schema, which contains a table named 'corona\_analysis' with columns: Province, Country\_Reg, Latitude, Longitude, Date\_s, and Confirmed. The 'Query 1' editor in the center contains the following SQL query:

```
-- Q14. Find Country having highest number of the Confirmed case
SELECT
  `Country_Region` AS country,
  MAX(Confirmed) AS highest_confirmed_cases
FROM
  corona_analysis
GROUP BY
  country
ORDER BY
  highest_confirmed_cases DESC
LIMIT 1;
```

The 'Result Grid' at the bottom shows the output of the query:

country	highest_confirmed_cases
Turkey	823225

## Q15. FIND COUNTRY HAVING LOWEST NUMBER OF THE DEATH CASES.

The screenshot shows a SQL IDE interface with a menu bar (File, Edit, View, Query, Database, Server, Tools, Scripting, Help) and a toolbar. The left sidebar contains a 'Navigator' pane with 'SCHEMAS' and a search filter. The 'corona\_dataset' schema is expanded, showing 'Tables' and 'Columns'. The 'Columns' list includes Province, Country\_Reg, Latitude, Longitude, Date\_s, and Confirmed. The main editor displays 'Query 1' with the following SQL code:

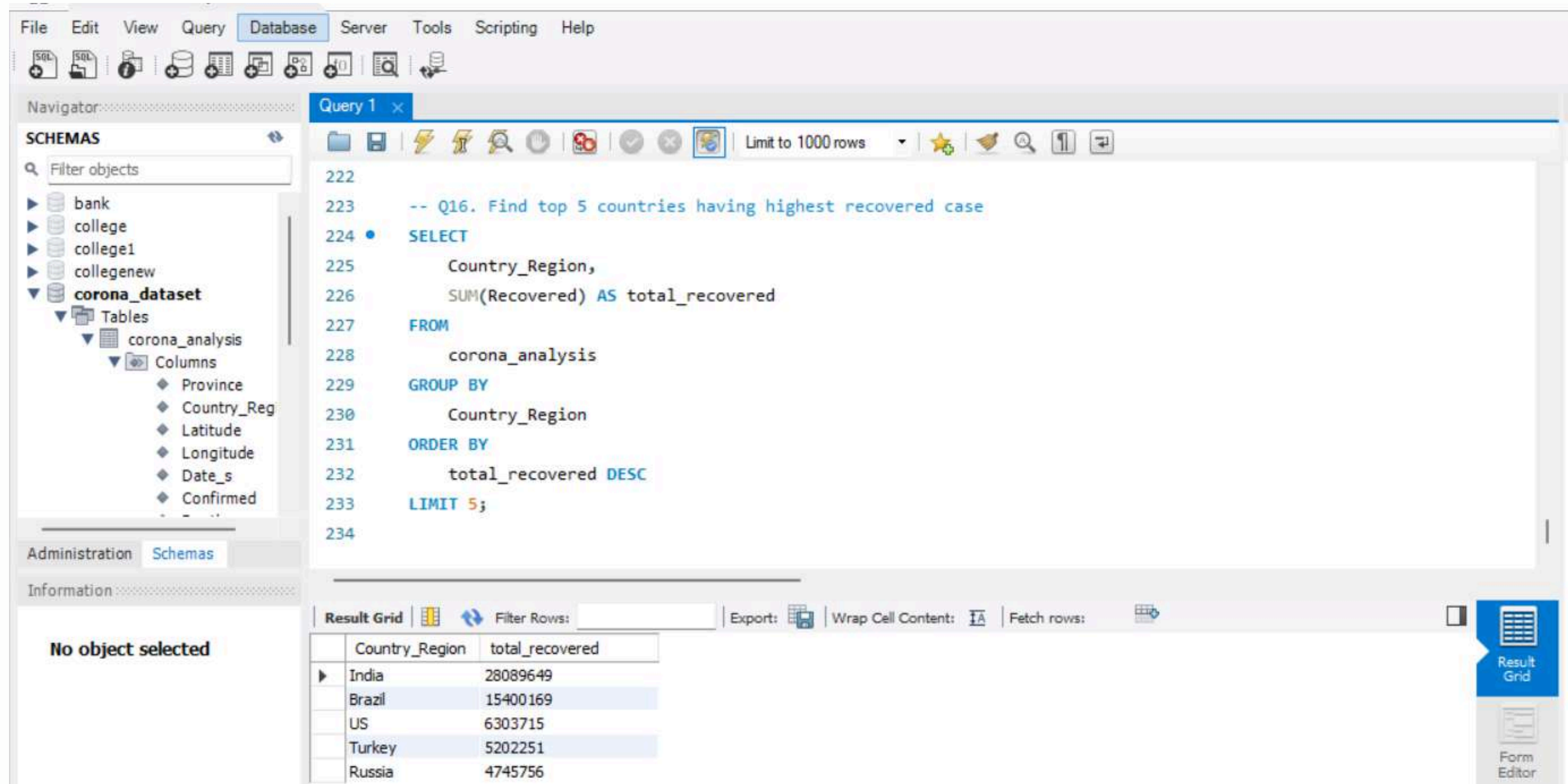
```
210
211 -- Q15. Find Country having lowest number of the death case
212 • SELECT
213     Country_Region,
214     SUM(Deaths) AS total_deaths
215 FROM
216     corona_analysis
217 GROUP BY
218     Country_Region
219 ORDER BY
220     total_deaths ASC
221 LIMIT 4;
```

The bottom of the IDE shows a 'Result Grid' with the following data:

Country_Region	total_deaths
Dominica	0
Samoa	0
Kiribati	0
Marshall Islands	0

The 'Information' pane on the left shows 'No object selected'. The bottom right of the IDE has buttons for 'Result Grid', 'Form Editor', and 'Fetch rows'.

## Q16. FIND TOP 5 COUNTRIES HAVING HIGHEST RECOVERED CASES.



The screenshot shows a database management interface with a menu bar (File, Edit, View, Query, Database, Server, Tools, Scripting, Help) and a toolbar. The left sidebar contains a 'Navigator' pane with 'SCHEMAS' and a search filter. The 'corona\_dataset' schema is expanded, showing 'Tables' and 'corona\_analysis' with its columns: Province, Country\_Reg, Latitude, Longitude, Date\_s, and Confirmed. The main area displays 'Query 1' with the following SQL code:

```
-- Q16. Find top 5 countries having highest recovered case
SELECT
    Country_Region,
    SUM(Recovered) AS total_recovered
FROM
    corona_analysis
GROUP BY
    Country_Region
ORDER BY
    total_recovered DESC
LIMIT 5;
```

The bottom of the interface shows the 'Result Grid' with the following data:

Country_Region	total_recovered
India	28089649
Brazil	15400169
US	6303715
Turkey	5202251
Russia	4745756





THANK YOU!

IT'S Q & A TIME!

