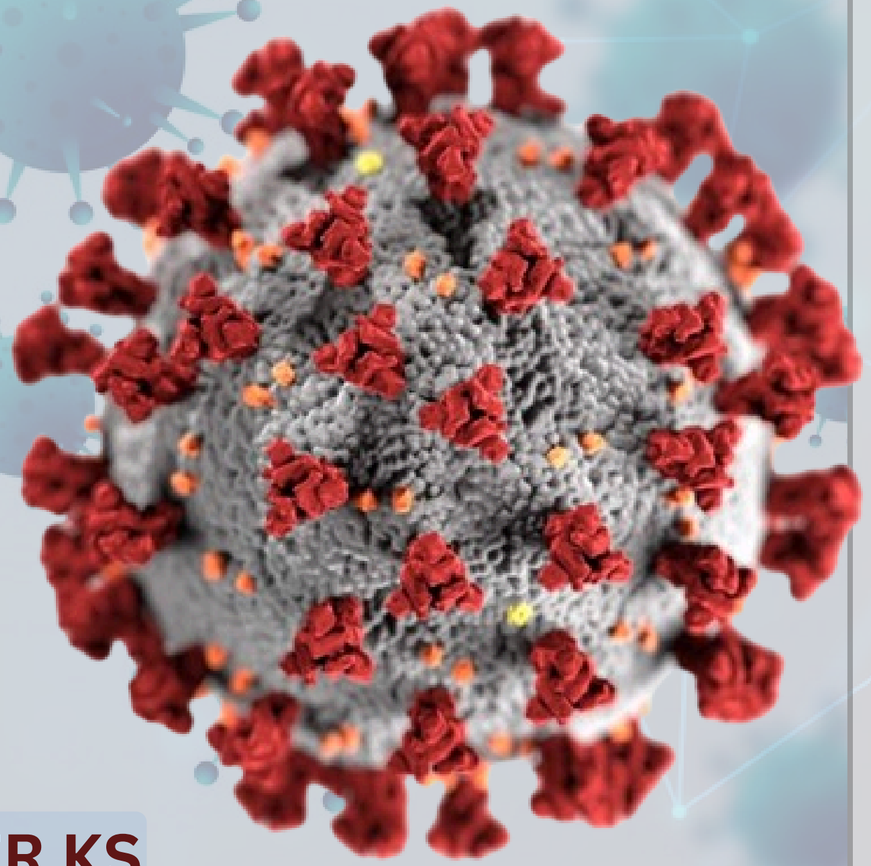
# COVID-19 ANALYSIS USING SQL

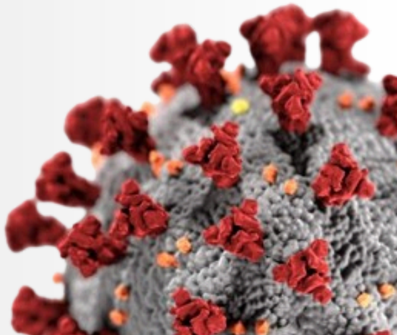


MAHAMMAD BASHEER KS

@MENTORNESS

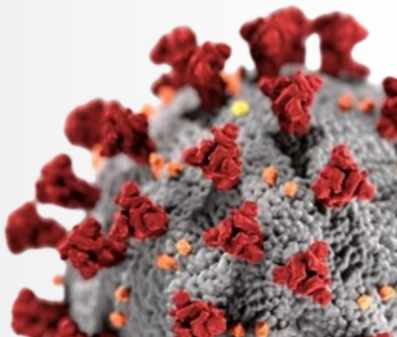
# Introduction


This project aims to analyze COVID-19 data using SQL to gain insights into the patterns, trends, and factors influencing the transmission and severity of the virus.



# OBJECTIVES

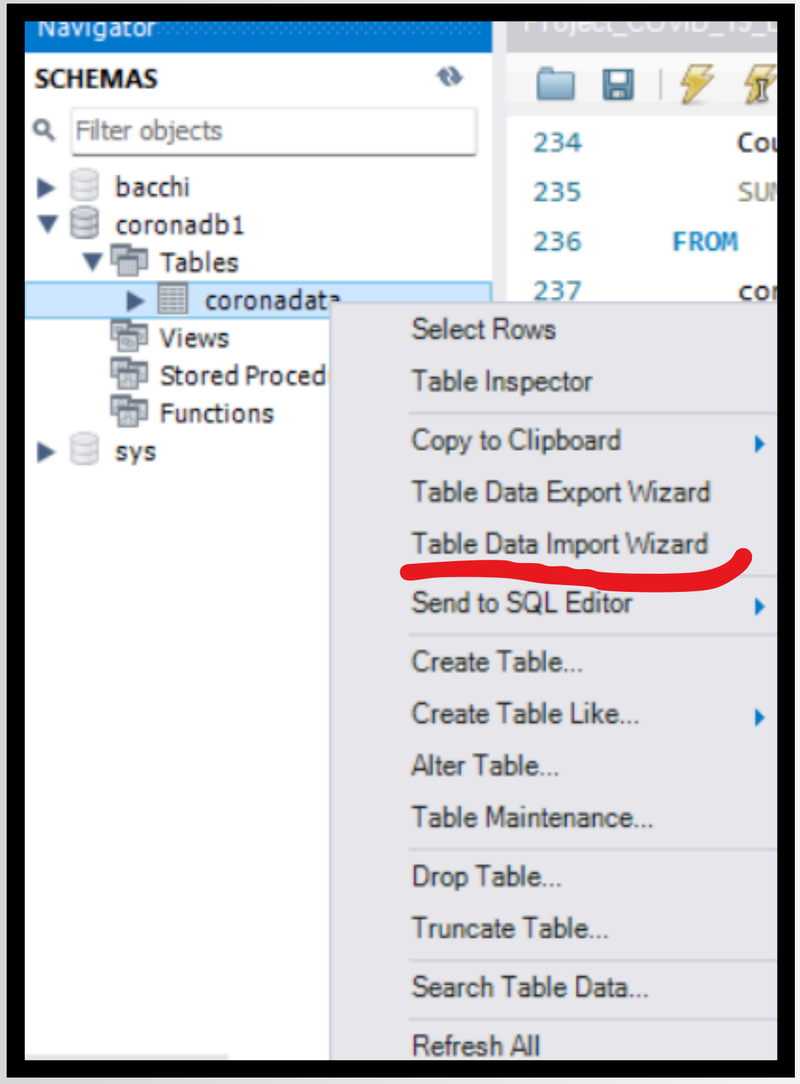
- Retrieve and preprocess COVID-19 data from the given dataset CoronaVirusDataset.csv
- The dataset contains data from 121 countries, and we are analyzing it using SQL.
- MySQL Workbench is being utilized for the project.
- Writing queries to the 16 questions from Mentorness.



# Loading Data to MySQL

## Steps:

1. Connect to your MySQL database server.
2. Create a new schema in Navigator.
4. Right click on schema and Select ‘Table Data Import Wizard’.
5. Browse the .csv file in the PC, containing the data to work.
6. Click ‘OK’, and finish.




# Data PreProcessing

Let's look at Dataset by Retrieving all the data.

```
SELECT * FROM coronadb1.coronadata;
```

```
6
7     -- Let's look at Dataset by Retrieving all the data.
8 •   SELECT * FROM coronadb1.coronadata;
9
10
11
12
13
14
15
16
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

| Province | country | Latitude | Longitude | Date | Confirmed | Deaths | Recovered |
|---|---|---|---|---|---|---|---|
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-22 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-23 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-24 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-25 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-26 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-27 | 0 | 0 | 0 |

coronadata 58

# Data PreProcessing

-- Converting the date column datatype from string to Date datatype.

```
UPDATE coronadata
SET Date = DATE_FORMAT(STR_TO_DATE(Date, '%d-%m-%Y'), '%Y-%m-%d');
```

# Data PreProcessing

-- Changing the column name from 'Country/Region' to 'country' to avoid potential errors
-- caused by the '/' symbol in the column name.

```
ALTER TABLE coronadb1.coronadata
CHANGE COLUMN `Country/Region` country
VARCHAR(255);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

| Province | country | Latitude | Longitude | Date | Confirmed | Deaths | Recovered |
|---|---|---|---|---|---|---|---|
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-22 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-23 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-24 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-25 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-26 | 0 | 0 | 0 |
| Afghanistan | Afghanistan | 33.93911 | 67.709953 | 2020-01-27 | 0 | 0 | 0 |

coronadata 58

# Data PreProcessing

-- The number of distinct Countries with coronavirus data.

```
21    -- The number of distinct Countries with coronavirus data.
22    SELECT COUNT(DISTINCT country) AS No_of_Countries FROM coronadb1.coronadata;
23    -- (There are records for 121 countries in the dataset.)
24
25
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| No_of_Countries |
| --- |
| 121 |

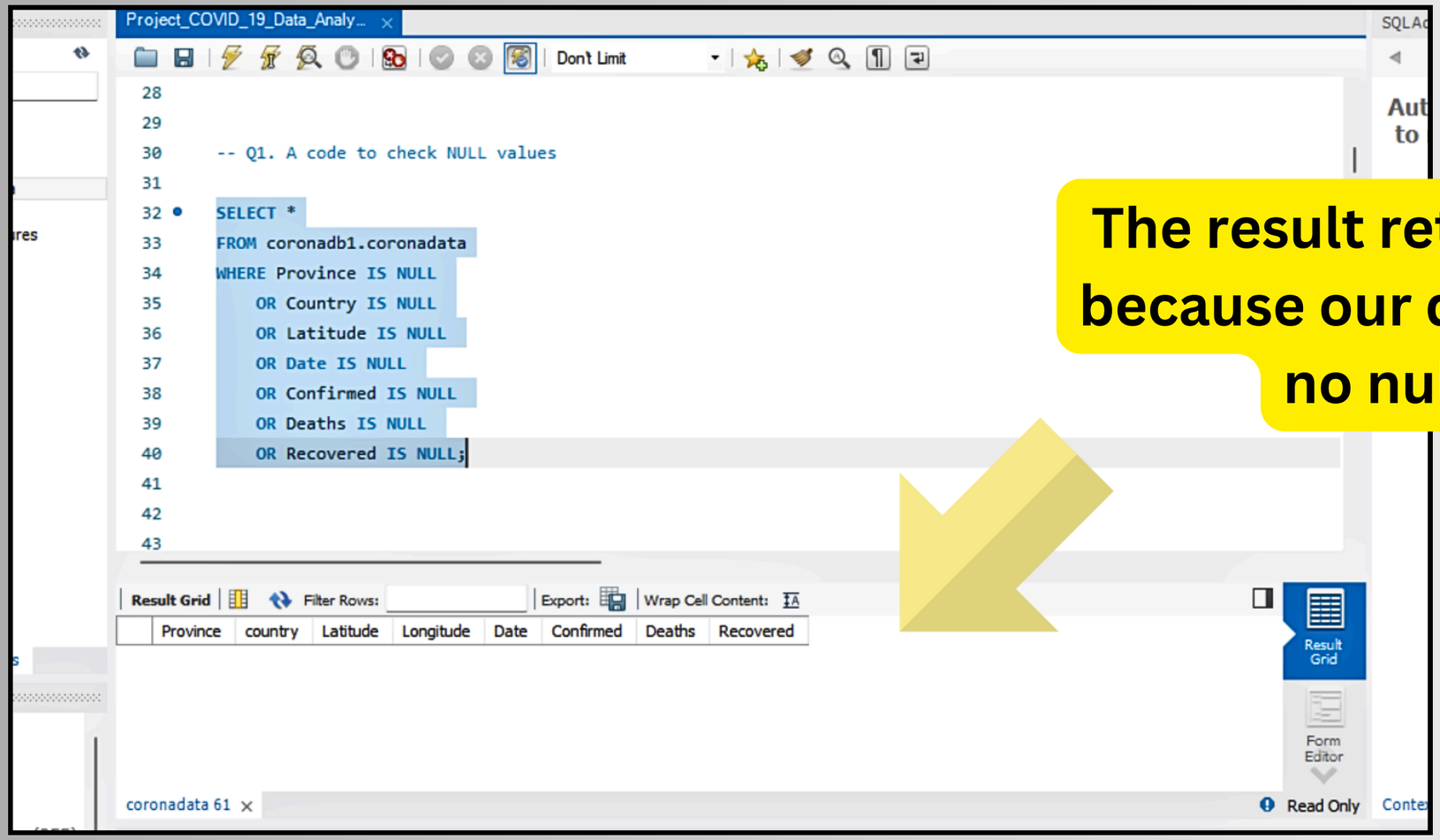
Result Grid

Form Editor

**There are records for 121 countries in the dataset**

# SQL QUERY

## -- Q1. Write a code to check NULL values




**The result return here blank, because our dataset contains no null values.**

# SQL QUERY

## -- Q2. If NULL values are present, update them with zeros for all columns.

```
42
43
44      -- Q2. If NULL values are present, update them with zeros for all columns.
45
46 •    UPDATE coronadb1.coronadata
47      SET Confirmed = CASE WHEN Confirmed IS NULL THEN 0 ELSE Confirmed END,
48          Deaths = CASE WHEN Deaths IS NULL THEN 0 ELSE Deaths END,
49          Recovered = CASE WHEN Recovered IS NULL THEN 0 ELSE Recovered END;
50
51      -- (None of Rows affected, because no Null values found)
52
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Province | country | Latitude | Longitude | Date | Confirmed | Deaths | Recovered |
|---|---|---|---|---|---|---|---|

Form Editor

coronadata 61 ×

Read Only

**None of Rows affected, because no Null values found**

# SQL QUERY

## -- Q3. check total number of rows

```
53
54      -- Q3. check total number of rows
55      SELECT COUNT(*)
56      FROM coronadb1.coronadata;
57
58      -- (Total No. of Rows 78386)
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| COUNT(*) |
|---|
| 78386 |

Result 62

**Total No. of Rows 78386**

# SQL QUERY

## -- Q4. Check what is start_date and end_date

```
60
61      -- Q4. Check what is start_date and end_date
62
63 •    SELECT
64          MIN(Date) AS Start_Date,
65          MAX(Date) AS End_Date
66      FROM coronadb1.coronadata;
67
68      -- (Start_Date: 2020-01-22, End_Date: 2021-06-13)
69
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Start_Date | End_Date |
|---|---|
| 2020-01-22 | 2021-06-13 |

Result Grid

Form Editor

Result 63 ×

Read Only

Output

**Start_Date: 2020-01-22,**
**End_Date: 2021-06-13**

# SQL QUERY

## -- Q5. Number of month present in dataset

```
71      -- Q5. Number of month present in dataset
72
73  •   SELECT
74          COUNT(DISTINCT DATE_FORMAT(Date, '%Y-%m')) AS Num_Months
75      FROM coronadb1.coronadata;
76
77      -- (Num of Months: 18)
78
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Num_Months |
|---|
| 18 |

**Num of Months: 18**

Result Grid | Form Editor | Result 64 | Read Only

# SQL QUERY

## -- Q6. Find monthly average for confirmed, deaths, recovered

```
82 •  SELECT
83        YEAR(Date) AS Year,
84        MONTH(Date) AS Month,
85        AVG(Confirmed) AS Avg_Confirmed,
86        AVG(Deaths) AS Avg_Deaths,
87        AVG(Recovered) AS Avg_Recovered
88    FROM coronadb1.coronadata
89    GROUP BY YEAR(Date), MONTH(Date);
90
91
92
```

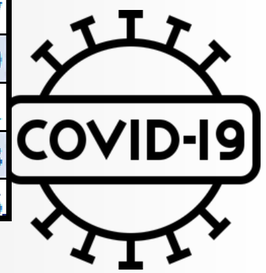
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Year | Month | Avg_Confirmed | Avg_Deaths | Avg_Recovered |
|---|---|---|---|---|
| 2020 | 1 | 4.1455 | 0.1234 | 0.0929 |
| 2020 | 2 | 15.2960 | 0.5936 | 7.0320 |
| 2020 | 3 | 161.1303 | 8.6607 | 27.8739 |
| 2020 | 4 | 505.8004 | 41.5223 | 171.6422 |
| 2020 | 5 | 574.8498 | 30.2809 | 318.2964 |
| 2020 | 6 | 859.2281 | 29.8175 | 548.7916 |
| 2020 | 7 | 1432.3611 | 35.1096 | 983.0582 |

Result 70

### Output

| Year | Month | Avg_Confirmed | Avg_Deaths | Avg_Recovered |
|---|---|---|---|---|
| 2020 | 1 | 4.1455 | 0.1234 | 0.0929 |
| 2020 | 2 | 15.296 | 0.5936 | 7.032 |
| 2020 | 3 | 161.1303 | 8.6607 | 27.8739 |
| 2020 | 4 | 505.8004 | 41.5223 | 171.6422 |
| 2020 | 5 | 574.8498 | 30.2809 | 318.2964 |
| 2020 | 6 | 859.2281 | 29.8175 | 548.7916 |
| 2020 | 7 | 1432.3611 | 35.1096 | 983.0582 |
| 2020 | 8 | 1611.8429 | 37.5367 | 1299.2947 |
| 2020 | 9 | 1784.5874 | 34.7773 | 1438.9067 |
| 2020 | 10 | 2412.1996 | 36.7583 | 1420.6431 |
| 2020 | 11 | 3592.1944 | 56.7634 | 1985.3446 |
| 2020 | 12 | 4050.4397 | 71.2183 | 2497.885 |
| 2021 | 1 | 3911.2285 | 84.1837 | 1919.637 |
| 2021 | 2 | 2433.3636 | 69.1649 | 1558.3917 |
| 2021 | 3 | 2916.7972 | 59.1998 | 1652.2859 |
| 2021 | 4 | 4699.3552 | 78.4387 | 3074.7851 |
| 2021 | 5 | 4005.2541 | 76.7803 | 4007.5078 |
| 2021 | 6 | 2508.6324 | 66.2622 | 2769.4496 |



# SQL QUERY

## Q7. Find most frequent value for confirmed, deaths, recovered each month

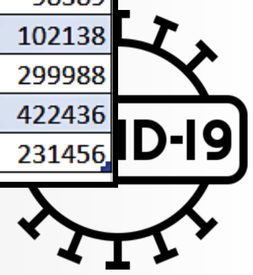
```
95 •  SELECT
96        Year(Date) as Year,
97        Month(Date) as Month,
98        SUBSTRING_INDEX(GROUP_CONCAT(confirmed ORDER BY confirmed DESC),',',1) AS Most_Frequent_Confirmed,
99        SUBSTRING_INDEX(GROUP_CONCAT(Deaths ORDER BY Deaths DESC),',',1) AS Most_Frequent_Deaths,
100       SUBSTRING_INDEX(GROUP_CONCAT(Recovered ORDER BY Recovered DESC),',',1) AS Most_Frequent_Recovered
101   FROM
102       coronadb1.coronadata
103   GROUP BY
104       Month,Year
105   ORDER BY
106       Year,Month;
107
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Year | Month | Most_Frequent_Confirmed | Most_Frequent_Deaths | Most_Frequent_Recovered |
|---|---|---|---|---|
| 2020 | 1 | 2131 | 49 | 51 |
| 2020 | 2 | 14840 | 242 | 3418 |
| 2020 | 3 | 26314 | 1085 | 4289 |
| 2020 | 4 | 50740 | 2607 | 33227 |
| 2020 | 5 | 34907 | 2309 | 51717 |
| 2020 | 6 | 54771 | 2003 | 94305 |
| 2020 | 7 | 75866 | 1595 | 140050 |

**Output**

| Year | Month | Most_Frequent_Confirmed | Most_Frequent_Deaths | Most_Frequent_Recovered |
|---|---|---|---|---|
| 2020 | 1 | 2131 | 49 | 51 |
| 2020 | 2 | 14840 | 242 | 3418 |
| 2020 | 3 | 26314 | 1085 | 4289 |
| 2020 | 4 | 50740 | 2607 | 33227 |
| 2020 | 5 | 34907 | 2309 | 51717 |
| 2020 | 6 | 54771 | 2003 | 94305 |
| 2020 | 7 | 75866 | 1595 | 140050 |
| 2020 | 8 | 85687 | 1505 | 95881 |
| 2020 | 9 | 97894 | 1703 | 101468 |
| 2020 | 10 | 99264 | 3351 | 388340 |
| 2020 | 11 | 207933 | 2259 | 139292 |
| 2020 | 12 | 823225 | 3752 | 1123456 |
| 2021 | 1 | 300462 | 4475 | 87090 |
| 2021 | 2 | 134975 | 3907 | 98389 |
| 2021 | 3 | 100158 | 3869 | 102138 |
| 2021 | 4 | 401993 | 4249 | 299988 |
| 2021 | 5 | 414188 | 4529 | 422436 |
| 2021 | 6 | 134154 | 7374 | 231456 |



# SQL QUERY

## -- Q8. Find minimum values for confirmed, deaths, recovered per year

```
112 •   SELECT
113         Year(Date) as Year,
114         MIN(Confirmed) as Min_Confirmed,
115         MIN(Deaths) as Min_Deaths,
116         MIN(Recovered) as Min_Recovered
117     FROM
118         coronadb1.coronadata
119     GROUP BY
120         Year;
121
122
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| | Year | Min_Confirmed | Min_Deaths | Min_Recovered |
|---|---|---|---|---|
| ▶ | 2020 | 0 | 0 | 0 |
| | 2021 | 0 | 0 | 0 |

**Output**

| Year | Min_Confirmed | Min_Deaths | Min_Recovered |
|---|---|---|---|
| 2020 | 0 | 0 | 0 |
| 2021 | 0 | 0 | 0 |

# SQL QUERY

## -- Q9. Find maximum values of confirmed, deaths, recovered per year

```
126 •   SELECT
127         Year(Date) as Year,
128         MAX(Confirmed) as Max_Confirmed,
129         MAX(Deaths) as Max_Deaths,
130         MAX(Recovered) as Max_Recovered
131     FROM
132         coronadb1.coronadata
133     GROUP BY
134         Year;
135
136
137
```

Result Grid | Filter Rows: | Export:

| Year | Max_Confirmed | Max_Deaths | Max_Recovered |
|---|---|---|---|
| 2020 | 823225 | 3752 | 1123456 |
| 2021 | 414188 | 7374 | 422436 |

**Output**

| Year | Max_Confirmed | Max_Deaths | Max_Recovered |
|---|---|---|---|
| 2020 | 823225 | 3752 | 1123456 |
| 2021 | 414188 | 7374 | 422436 |

# SQL QUERY

## Q10. The total number of case of confirmed, deaths, recovered each month

```
140 •   SELECT
141         Year(Date) as Year,
142         Month(Date) as Month,
143         SUM(Confirmed) as Total_Confirmed,
144         SUM(Deaths) AS Total_Deaths,
145         SUM(Recovered) AS Total_Recovered
146     FROM
147         coronadb1.coronadata
148     GROUP BY
149         Year,Month;
```

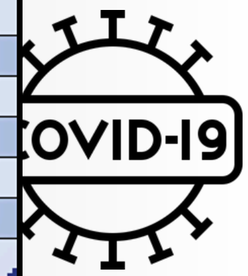
Result Grid | Filter Rows:

| Year | Month | Total_Confirmed | Total_Deaths |
|---|---|---|---|
| 2020 | 1 | 6384 | 190 |
| 2020 | 2 | 68312 | 2651 |
| 2020 | 3 | 769236 | 41346 |
| 2020 | 4 | 2336798 | 191833 |
| 2020 | 5 | 2744333 | 144561 |
| 2020 | 6 | 3969634 | 137757 |
| 2020 | 7 | 6838092 | 167613 |
| 2020 | 8 | 7694938 | 179200 |

Result 76

**Output**

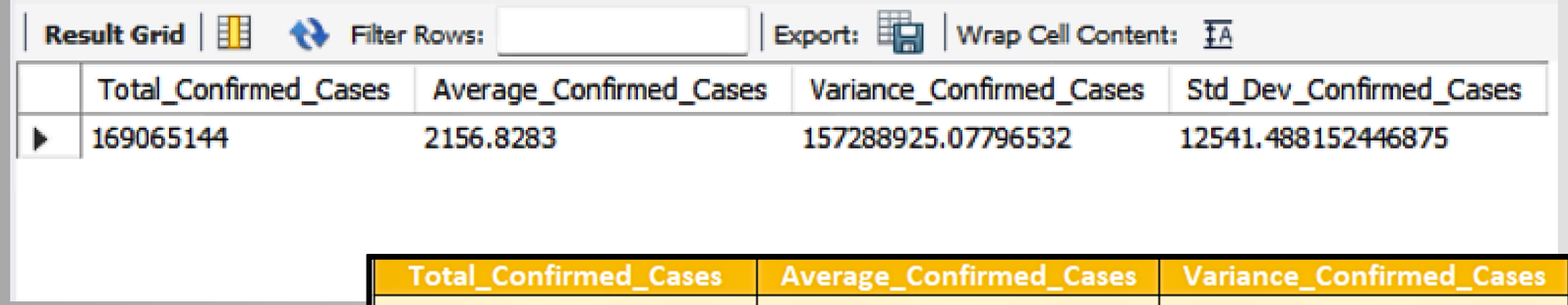
| Year | Month | Total_Confirmed | Total_Deaths | Total_Recovered |
|---|---|---|---|---|
| 2020 | 1 | 6384 | 190 | 143 |
| 2020 | 2 | 68312 | 2651 | 31405 |
| 2020 | 3 | 769236 | 41346 | 133070 |
| 2020 | 4 | 2336798 | 191833 | 792987 |
| 2020 | 5 | 2744333 | 144561 | 1519547 |
| 2020 | 6 | 3969634 | 137757 | 2535417 |
| 2020 | 7 | 6838092 | 167613 | 4693120 |
| 2020 | 8 | 7694938 | 179200 | 6202833 |
| 2020 | 9 | 8244794 | 160671 | 6647749 |
| 2020 | 10 | 11515841 | 175484 | 6782150 |
| 2020 | 11 | 16595938 | 262247 | 9172292 |
| 2020 | 12 | 19336799 | 339996 | 11924903 |
| 2021 | 1 | 18672205 | 401893 | 9164347 |
| 2021 | 2 | 10492664 | 298239 | 6719785 |
| 2021 | 3 | 13924790 | 282620 | 7888013 |
| 2021 | 4 | 21711021 | 362387 | 14205507 |
| 2021 | 5 | 19121083 | 366549 | 19131842 |
| 2021 | 6 | 5022282 | 132657 | 5544438 |



# SQL QUERY

## Q11. Check how corona virus spread out with respect to confirmed case (Eg.: total confirmed cases, their average, variance & STDEV )

```
156 •   SELECT
157         SUM(Confirmed) AS Total_Confirmed_Cases,
158         AVG(Confirmed) AS Average_Confirmed_Cases,
159         VARIANCE(Confirmed) AS Variance_Confirmed_Cases,
160         STDDEV(Confirmed) AS Std_Dev_Confirmed_Cases
161     FROM coronadb1.coronadata;
162
```




| Total_Confirmed_Cases | Average_Confirmed_Cases | Variance_Confirmed_Cases | Std_Dev_Confirmed_Cases |
|---|---|---|---|
| 169065144 | 2156.8283 | 157288925.07796532 | 12541.488152446875 |

| Total_Confirmed_Cases | Average_Confirmed_Cases | Variance_Confirmed_Cases | Std_Dev_Confirmed_Cases |
|---|---|---|---|
| 169065144 | 2156.8283 | 157288925.1 | 12541.48815 |

# SQL QUERY

## Q12. Check how corona virus spread out with respect to death case per month (Eg.: total confirmed cases, their average, variance & STDEV )

```
SELECT
    YEAR(Date) AS Year,
    MONTH(Date) AS Month,
    SUM(Deaths) AS Total_Death_Cases,
    AVG(Deaths) AS Average_Death_Cases,
    VARIANCE(Deaths) AS Variance_Death_Cases,
    STDDEV(Deaths) AS Std_Dev_Death_Cases
FROM coronadb1.coronadata
GROUP BY
    YEAR, MONTH;
```

### Output

| Year | Month | Total_Death_Cases | Average_Death_Cases | Variance_Death_Cases | Std_Dev_Death_Cases |
|---|---|---|---|---|---|
| 2020 | 1 | 190 | 0.1234 | 4.24581717 | 2.060538078 |
| 2020 | 2 | 2651 | 0.5936 | 68.32184882 | 8.265703166 |
| 2020 | 3 | 41346 | 8.6607 | 3900.792265 | 62.45632286 |
| 2020 | 4 | 191833 | 41.5223 | 40504.26812 | 201.2567219 |
| 2020 | 5 | 144561 | 30.2809 | 20684.91167 | 143.8225006 |
| 2020 | 6 | 137757 | 29.8175 | 16929.44571 | 130.1132034 |
| 2020 | 7 | 167613 | 35.1096 | 21140.15494 | 145.3965438 |
| 2020 | 8 | 179200 | 37.5367 | 23272.99646 | 152.5548965 |
| 2020 | 9 | 160671 | 34.7773 | 20102.76922 | 141.7842348 |
| 2020 | 10 | 175484 | 36.7583 | 17580.07102 | 132.5898602 |
| 2020 | 11 | 262247 | 56.7634 | 27773.7936 | 166.6547137 |
| 2020 | 12 | 339996 | 71.2183 | 65345.3692 | 255.6274031 |
| 2021 | 1 | 401893 | 84.1837 | 102758.4323 | 320.5595613 |
| 2021 | 2 | 298239 | 69.1649 | 68478.87147 | 261.6846795 |
| 2021 | 3 | 282620 | 59.1998 | 54385.9697 | 233.2079967 |
| 2021 | 4 | 362387 | 78.4387 | 94611.47092 | 307.589777 |
| 2021 | 5 | 366549 | 76.7803 | 131769.4693 | 363.0006464 |
| 2021 | 6 | 132657 | 66.2622 | 112963.673 | 336.1006888 |

# SQL QUERY

-- Q13. Check how corona virus spread out with respect to recovered case
--  (Eg.: total confirmed cases, their average, variance & STDEV )

```
SELECT
    SUM(Recovered) AS Total_Recovered_Cases,
    AVG(Recovered) AS Average_Recovered_Cases,
    VARIANCE(Recovered) AS Variance_Recovered_Cases,
    STDDEV(Recovered) AS Std_Dev_Recovered_Cases
FROM coronadb1.coronadata;
```

| Total_Recovered_Cases | Average_Recovered_Cases | Variance_Recovered_Cases | Std_Dev_Recovered_Cases |
|---|---|---|---|
| 113089548 | 1442.7264 | 107029523.3 | 10345.5074 |

# SQL QUERY

## -- Q14. Find Country having highest number of the Confirmed case

```
200 •   SELECT
201         Country,
202         SUM(Confirmed) AS Highest_Confirmed_Cases
203     FROM
204         coronadb1.coronadata
205     GROUP BY
206         Country
207     ORDER BY
208         Highest_Confirmed_Cases DESC
209     LIMIT 1;
210
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Country | Highest_Confirmed_Cases |
|---|---|
| US | 33461982 |

**The Country having highest number of the Confirmed case is United State with 33461982 cases.**

# SQL QUERY

## -- Q15. Find Country having lowest number of the death case

```
218 •   SELECT
219         Country,
220         SUM(Deaths) AS Lowest_Death_Cases
221     FROM
222         coronadb1.coronadata
223     GROUP BY
224         Country
225     ORDER BY
226         Lowest_Death_Cases ASC
227     LIMIT 5;
228
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| Country | Lowest_Death_Cases |
|---|---|
| Samoa | 0 |
| Dominica | 0 |
| Kiribati | 0 |
| Marshall Islands | 0 |
| Bhutan | 1 |


| Country | Lowest_Death_Cases |
|---|---|
| Samoa | 0 |
| Dominica | 0 |
| Kiribati | 0 |
| Marshall Islands | 0 |
| Bhutan | 1 |


**Note:Adjust the limit to get more countries list with lowest Death case.**

# SQL QUERY

## -- Q16. Find top 5 countries having highest recovered case

```
233 •   SELECT
234         Country,
235         SUM(Recovered) AS Total_Recovered_Cases
236     FROM
237         coronadb1.coronadata
238     GROUP BY
239         Country
240     ORDER BY
241         Total_Recovered_Cases DESC
242     LIMIT 5;
```

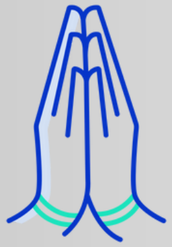
Result Grid | Filter Rows: | Export: | Wrap

| Country | Total_Recovered_Cases |
|---|---|
| India | 28089649 |
| Brazil | 15400169 |
| US | 6303715 |
| Turkey | 5202251 |
| Russia | 4745756 |

| Country | Total_Recovered_Cases |
|---|---|
| India | 28089649 |
| Brazil | 15400169 |
| US | 6303715 |
| Turkey | 5202251 |
| Russia | 4745756 |

# Conclusion Points

- Checked for NULL values in the dataset to ensure data integrity.
- Checked the total number of rows to understand the dataset's size and scope.
- Identified the start_date and end_date to establish the timeframe of the dataset.
- Calculated monthly averages and identified the most frequent values for confirmed cases, deaths, and recoveries to understand trends and recurring patterns over time.
- Identified country with highest confirmed cases, lowest death cases, and top 5 countries with highest recovered cases.




**MAHAMMAD BASHEER KS**

**@MENTORNESS**