

Naresh Kalluri (10755799)

Intro to Data Science (CSC 346 D01 Spring 2022).

Assignment – 1 (Project 2).

Date: 2-15-2022.

GitHub Repository: <https://github.com/nk755799/IDS>

## Project 2:

### Problem 1:

The dataset **USAArrests.csv** contains statistics in arrests per 100,000 residents for assault and murder, in each of the 50 US states, in 1973. Also given is the percentage of the population living in urban areas: [https://github.com/bforoura/IDS/tree/main/HW2 \(Links to an external site.\)](https://github.com/bforoura/IDS/tree/main/HW2)

Excel:

### Purpose of the Project:

To prepare the dataset for analysis by using the pre-processing Techniques.

### Operations:

A. By seeing the data, it can prove that there are some missing values in the data. The one missing value I found on that data is that Georgia doesn't have the Assault value.

By using the Mean(average) I can get that value, so I took the whole column for Mean and I got the value = 166. so, I placed 166 as the Assault value for the state of Georgia.

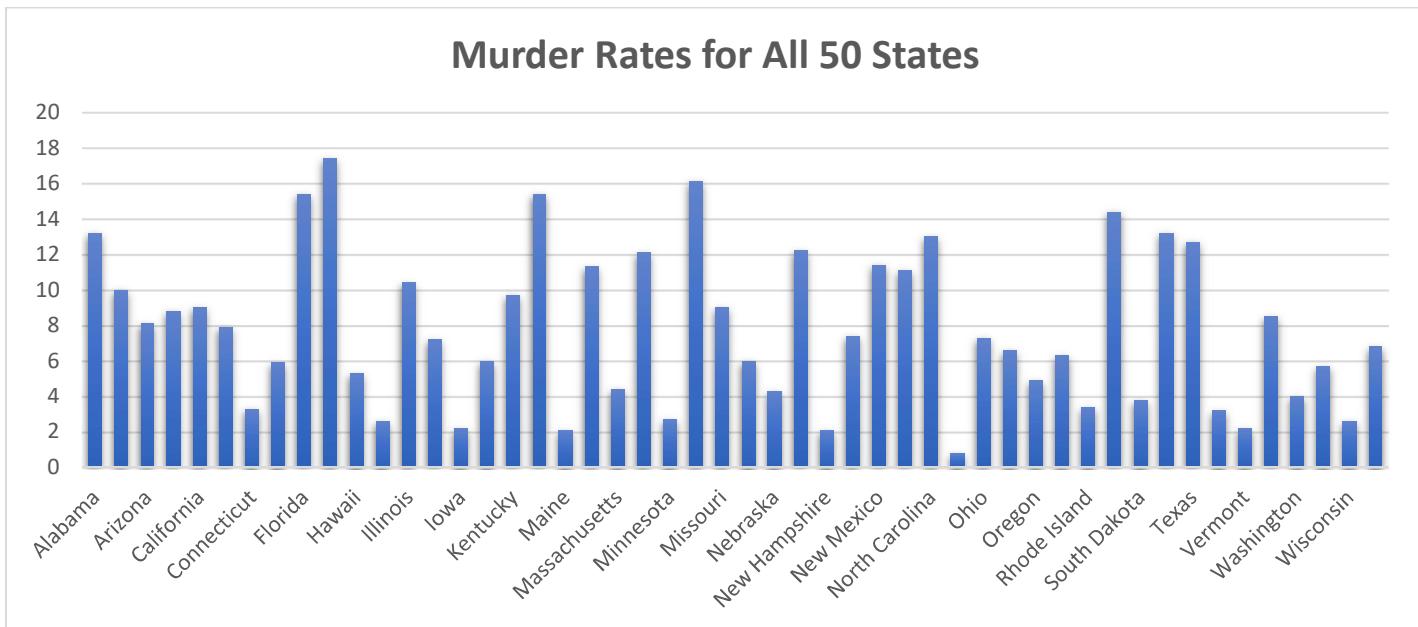
Result:

H10	A	B	C	D	E
1	State	Murder	Assault	UrbanPop	
2	Alabama	13.2	236	58	
3	Alaska	10	263	48	
4	Arizona	8.1	294	80	
5	Arkansas	8.8	190	50	
6	California	9	276	91	
7	Colorado	7.9	204	78	
8	Connecticut	3.3	110	77	
9	Delaware	5.9	238	72	
10	Florida	15.4	335	80	
11	Georgia	17.4	166	60	
12	Hawaii	5.3	46	83	
13	Idaho	2.6	120	54	
14	Illinois	10.4	249	83	
15	Indiana	7.2	113	65	
16	Iowa	2.2	56	57	
17	Kansas	6	115	66	
18	Kentucky	9.7	109	52	
19	Louisiana	15.4	249	66	
20	Maine	2.1	83	51	

B. There is only one missing value is there and I fixed that too.

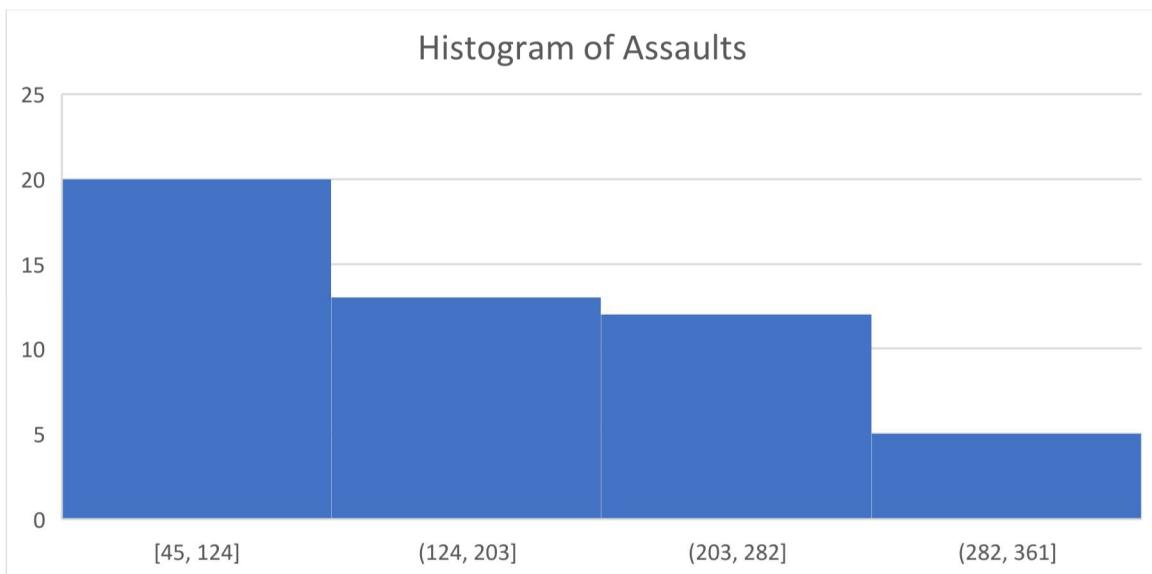
C. Plot Murder rates for all 50 states.

Result: For clearly view Please refer to my GitHub repository HW2 Folder



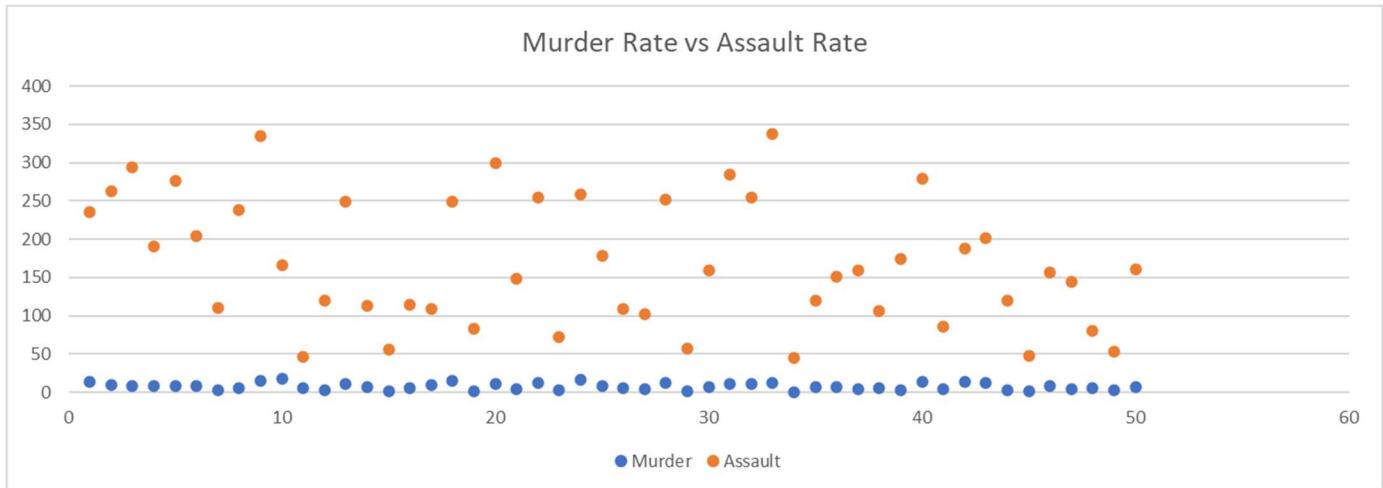
D. Plot histogram of assaults:

Result:



## E. Plot Murder rate vs. Assault rate.

Result:



## F. Prepare the dataset to establish a relationship between an urban population category and a crime type.

I converted the urban population into categories like Small, Medium, Large and Extra Large and got a relation.

	A	B	C	D	E	F	G	H	I	J	K	L
1	State	Murder	Assault	UrbanPop	Urban Category							
2	Alabama	13.2	236	58	Medium							
3	Alaska	10	263	48	Small							
4	Arizona	8.1	294	80	Extra Large							
5	Arkansas	8.8	190	50	Medium							
6	California	9	276	91	Extra Large							
7	Colorado	7.9	204	78	Extra Large							
8	Connecticut	3.3	110	77	Extra Large							
9	Delaware	5.9	238	72	Extra Large							
10	Florida	15.4	335	80	Extra Large							
11	Georgia	17.4	166	60	Large							
12	Hawaii	5.3	46	83	Extra Large							
13	Idaho	2.6	120	54	Medium							
14	Illinois	10.4	249	83	Extra Large							
15	Indiana	7.2	113	65	Large							
16	Iowa	2.2	56	57	Medium							
17	Kansas	6	115	66	Large							
18	Kentucky	9.7	109	52	Medium							

Row Labels	Sum of Murder	Sum of Assault	Count of UrbanPop
Extra Large	169.8	4150	21
Large	93.7	1797	12
Medium	59.9	1148	9
Small	66	1398	8
<b>Grand Total</b>	<b>389.4</b>	<b>8493</b>	<b>50</b>

MySQL:

A. Import the original CSV file into MySQL and create the table USArrests.

Result: Imported the original CSV file into MySQL workspace and created the table USArrests.

The screenshot shows the MySQL Workbench interface. In the Navigator pane, under the 'Tables' section of the 'ids20db' schema, the 'USArrests' table is selected. The 'Query 1' editor contains the following SQL code:

```

1 • use ids20db;
2 • select *
3   from USArrests
  
```

The Result Grid pane displays the data from the USArrests table:

State	Murder	Assault	UrbanPop
Connecticut	3.3	110	77
Delaware	5.9	238	72
Florida	15.4	335	80
Georgia	17.4	0	60
Hawaii	5.3	46	83
Idaho	2.6	120	54

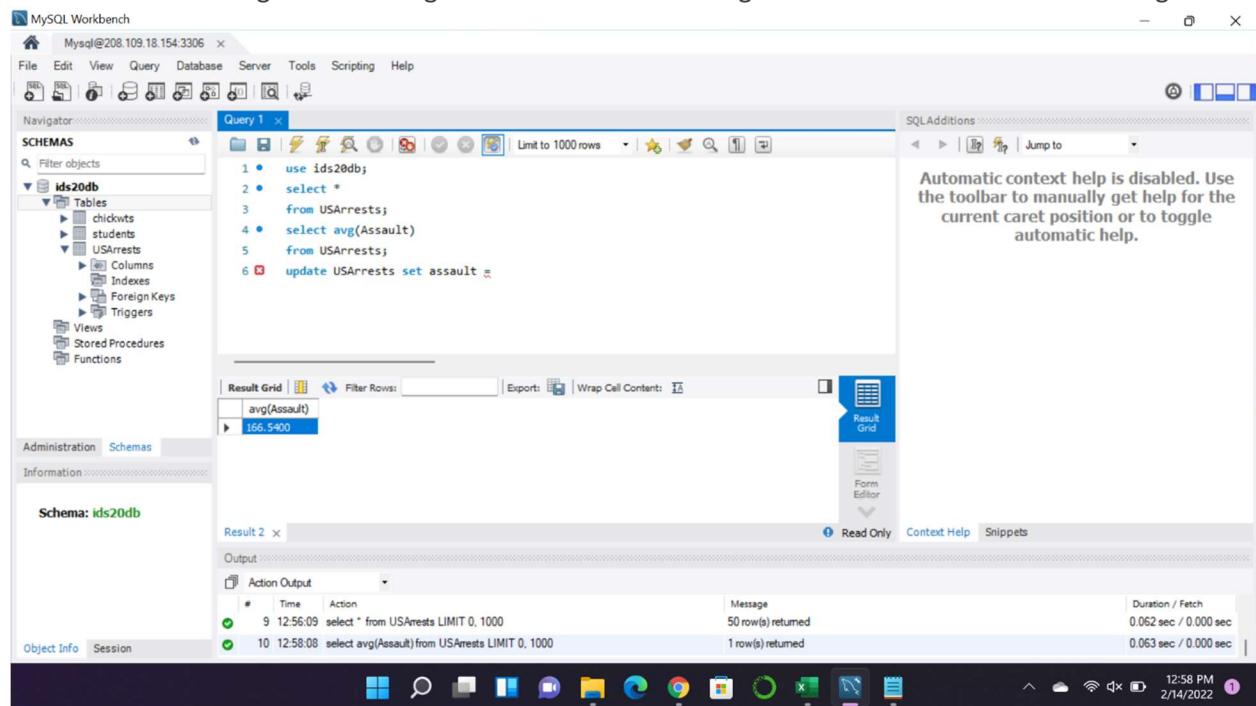
The Output pane shows the execution history:

- Action 8: 12:55:20 use ids20db - Message: 0 row(s) affected - Duration / Fetch: 0.078 sec
- Action 9: 12:56:09 select \* from USArrests LIMIT 0, 1000 - Message: 50 row(s) returned - Duration / Fetch: 0.062 sec / 0.000 sec

B. Use SQL to replace all missing values in a column by the average. Hint: Use the Update command.

Result:

First I took the average method to get the value of the missing Assault value of the state of Georgia.



The screenshot shows the MySQL Workbench interface. In the top-left pane, the Navigator displays the schema 'ids20db' with its tables: chickwts, students, and USArests. The USArests table has columns: Murder, Assault, and UrbanPop. In the top-right pane, the SQL Editor contains the following code:

```
1 • use ids20db;
2 • select *
3   from USArests;
4 • select avg(Assault)
5   from USArests;
6 • update USArests set assault =
```

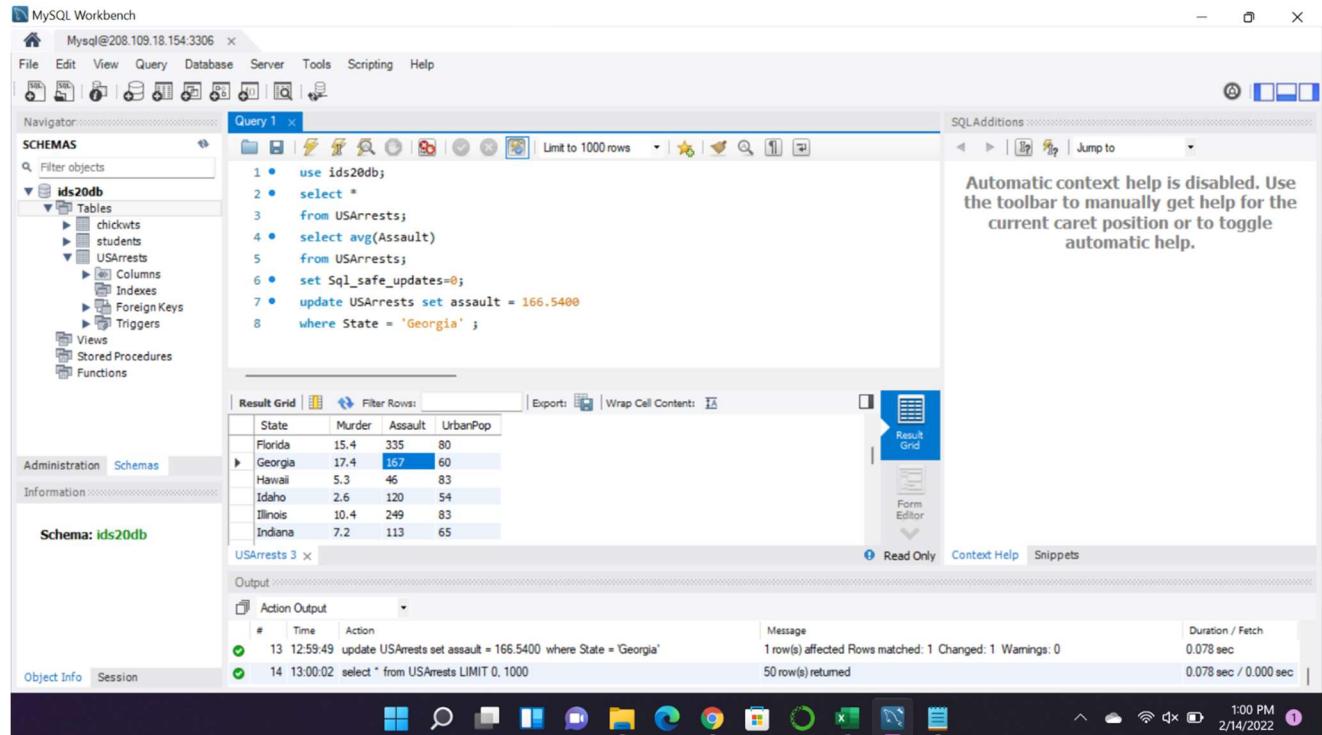
The Result Grid shows the output of the first two queries:

	avg(Assault)
	166.5400

In the bottom-right pane, the Output window shows the execution of the update command:

#	Time	Action	Message	Duration / Fetch
9	12:56:09	select * from USArests LIMIT 0, 1000	50 row(s) returned	0.062 sec / 0.000 sec
10	12:58:08	select avg(Assault) from USArests LIMIT 0, 1000	1 row(s) returned	0.063 sec / 0.000 sec

Next used the Update command to update the Georgia Assault Value with an average of column '166.54'



The screenshot shows the MySQL Workbench interface. In the top-left pane, the Navigator displays the schema 'ids20db' with its tables: chickwts, students, and USArests. The USArests table has columns: Murder, Assault, and UrbanPop. In the top-right pane, the SQL Editor contains the following code:

```
1 • use ids20db;
2 • select *
3   from USArests;
4 • select avg(Assault)
5   from USArests;
6 • set Sql_safe_updates=0;
7 • update USArests set assault = 166.5400
8   where State = 'Georgia' ;
```

The Result Grid shows the updated data for the USArests table:

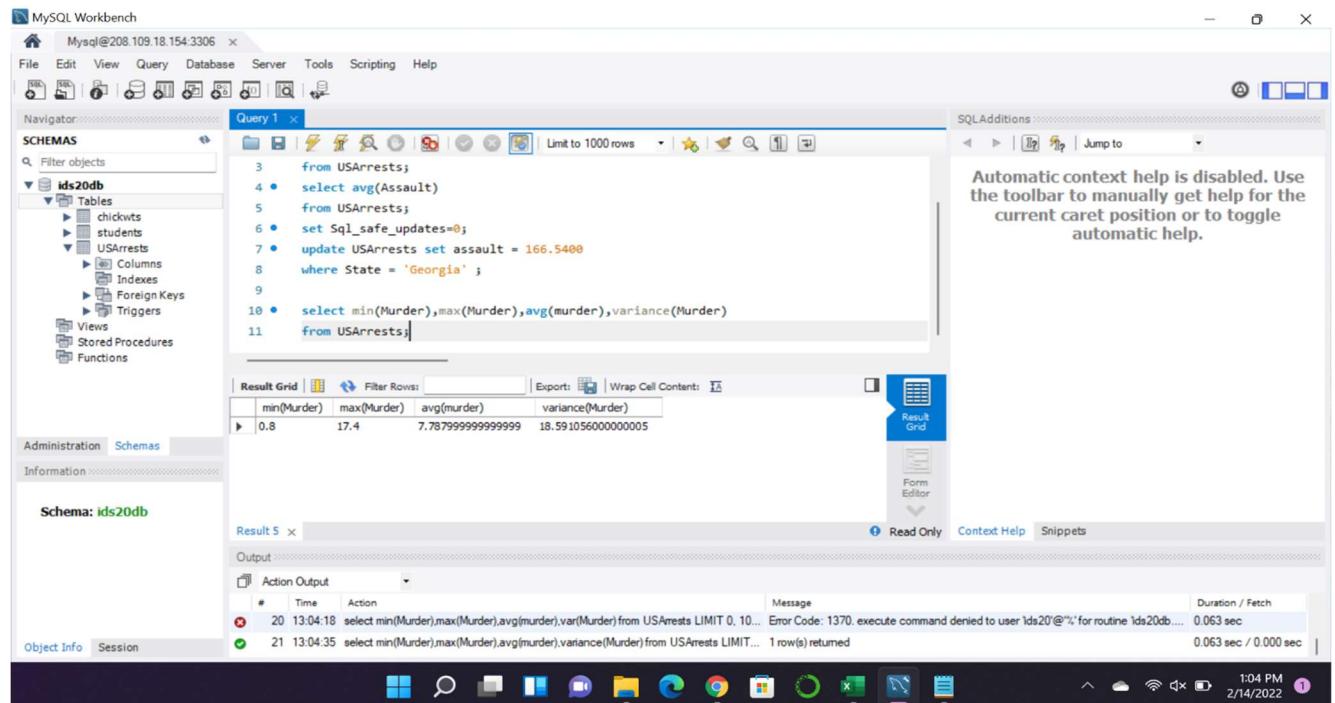
State	Murder	Assault	UrbanPop
Florida	15.4	335	80
Georgia	17.4	167	60
Hawaii	5.3	46	83
Idaho	2.6	120	54
Illinois	10.4	249	83
Indiana	7.2	113	65

In the bottom-right pane, the Output window shows the execution of the update command:

#	Time	Action	Message	Duration / Fetch
13	12:59:49	update USArests set assault = 166.5400 where State = 'Georgia'	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.078 sec
14	13:00:02	select * from USArests LIMIT 0, 1000	50 row(s) returned	0.078 sec / 0.000 sec

C. Find min, max, mean, and variance of all numeric attributes in SQL.

Result: For Murder:



The screenshot shows the MySQL Workbench interface with the following details:

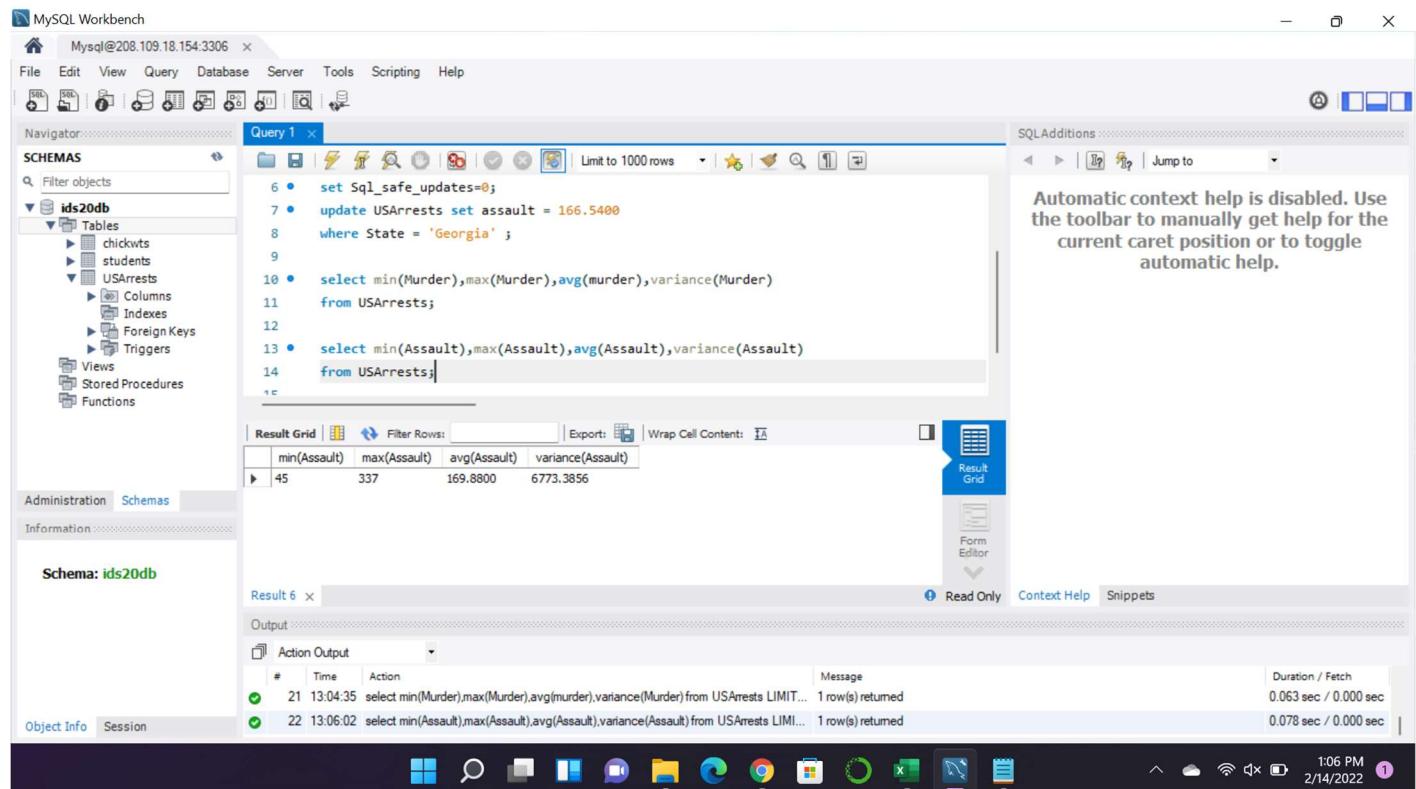
- Navigator:** Shows the schema **ids20db** with tables **chickwts**, **students**, and **USArests**.
- Query 1:** Contains the following SQL code:

```
3   from USArests;
4 •   select avg(Assault)
5   from USArests;
6 •   set Sql_safe_updates=0;
7 •   update USArests set assault = 166.5400
8   where State = 'Georgia';
9
10 •  select min(Murder),max(Murder),avg(murder),variance(Murder)
11   from USArests;
```
- Result Grid:** Displays the results of the last query:

min(Murder)	max(Murder)	avg(murder)	variance(Murder)
0.8	17.4	7.787999999999999	18.591056000000005
- SQLAdditions:** A note states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."
- Result 5:** Shows the log output for the executed queries:

#	Time	Action	Message	Duration / Fetch
20	13:04:18	select min(Murder),max(Murder),avg(murder),var(Murder) from USArests LIMIT 0, 10...	Error Code: 1370. execute command denied to user 'ids20'@'%' for routine 'ids20db....'	0.063 sec
21	13:04:35	select min(Murder),max(Murder),avg(murder),variance(Murder) from USArests LIMIT...	1 row(s) returned	0.063 sec / 0.000 sec
- Object Info:** Shows the schema **ids20db**.

For Assaults:



The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the schema **ids20db** with tables **chickwts**, **students**, and **USArests**.
- Query 1:** Contains the following SQL code:

```
6 •   set Sql_safe_updates=0;
7 •   update USArests set assault = 166.5400
8   where State = 'Georgia';
9
10 •  select min(Murder),max(Murder),avg(murder),variance(Murder)
11   from USArests;
12
13 •  select min(Assault),max(Assault),avg(Assault),variance(Assault)
14   from USArests;
```
- Result Grid:** Displays the results of the last query:

min(Assault)	max(Assault)	avg(Assault)	variance(Assault)
45	337	169.8800	6773.3856
- SQLAdditions:** A note states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."
- Result 6:** Shows the log output for the executed queries:

#	Time	Action	Message	Duration / Fetch
21	13:04:35	select min(Murder),max(Murder),avg(murder),variance(Murder) from USArests LIMIT...	1 row(s) returned	0.063 sec / 0.000 sec
22	13:06:02	select min(Assault),max(Assault),avg(Assault),variance(Assault) from USArests LIMI...	1 row(s) returned	0.078 sec / 0.000 sec
- Object Info:** Shows the schema **ids20db**.

## For Urban Population

The screenshot shows the MySQL Workbench interface with three queries in the Query Editor:

```
9
10 • select min(Murder),max(Murder),avg(murder),variance(Murder)
11   from USArests;
12
13 • select min(Assault),max(Assault),avg(Assault),variance(Assault)
14   from USArests;
15
16 • select min(UrbanPop),max(UrbanPop),avg(UrbanPop),variance(UrbanPop)
17   from USArests;
```

The Result Grid shows the following data:

min(UrbanPop)	max(UrbanPop)	avg(UrbanPop)	variance(UrbanPop)
32	91	65.5400	205.32839999999996

The SQLAdditions panel displays a message: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

D. Which state has the maximum murder rate?

Result:

The screenshot shows the MySQL Workbench interface with the following query in the Query Editor:

```
12
13 • select min(Assault),max(Assault),avg(Assault),variance(Assault)
14   from USArests;
15
16 • select min(UrbanPop),max(UrbanPop),avg(UrbanPop),variance(UrbanPop)
17   from USArests;
18
19 • select Max(Murder)
20   from USArests;
```

The Result Grid shows the following data:

Max(Murder)
17.4

The SQLAdditions panel displays a message: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

MySQL Workbench

Mysql@208.109.18.154:3306

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

ids20db

Tables

chickwts  
students  
USArests

Columns

State  
Murder  
Assault  
UrbanPop

Indexes  
Foreign Keys  
Triggers

Views  
Stored Procedures  
Functions

Administration Schemas

Information

Schema: ids20db

Query 1

```
14 from USArests;
15
16 • select min(UrbanPop),max(UrbanPop),avg(UrbanPop),variance(UrbanPop)
17 from USArests;
18
19 • select Max(Murder)
20 from USArests;
21
22 • select state
   from USArests;
```

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

state

Vermont  
West Virginia  
Mississippi  
North Dakota  
North Carolina  
South Dakota

USArests 10 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
25	13:09:49	select * from USArests order by UrbanPop LIMIT 0, 1000	50 row(s) returned	0.079 sec / 0.000 sec
26	13:10:19	select state from USArests order by UrbanPop LIMIT 0, 1000	50 row(s) returned	0.078 sec / 0.000 sec

Object Info Session

1:10 PM 2/14/2022

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

MySQL Workbench

Mysql@208.109.18.154:3306

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

ids20db

Tables

chickwts  
students  
USArests

Columns

State  
Murder  
Assault  
UrbanPop

Indexes  
Foreign Keys  
Triggers

Views  
Stored Procedures  
Functions

Administration Schemas

Information

Schema: ids20db

Query 1

```
21
22 • select state
23 from USArests
24 order by UrbanPop;
25
26
27 • select count(state)
28 from USArests
29 where Murder>8.1;
```

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

count(state)

21

Result 13 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
28	13:12:20	select * from USArests where Murder>8.1 LIMIT 0, 1000	21 row(s) returned	0.052 sec / 0.000 sec
29	13:12:45	select count(state) from USArests where Murder>8.1 LIMIT 0, 1000	1 row(s) returned	0.078 sec / 0.000 sec

Object Info Session

1:12 PM 2/14/2022

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

MySQL Workbench

Mysql@208.109.18.154:3306

File Edit View Query Database Server Tools Scripting Help

Navigator: Schemas

SCHEMAS: ids20db

Tables: chickwts, students, USArests

Columns: State, Murder, Assault, UrbanPop

Indexes, Foreign Keys, Triggers, Views, Stored Procedures, Functions

Administration, Schemas, Information

Schema: ids20db

Query 1

```
25
26
27 • select count(state)
28   from USArests
29   where Murder>8.1;
30
31 • select *
32   from USArests
33   where Murder>8.1;
34
```

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

State	Murder	Assault	UrbanPop
Alabama	13.2	236	58
Alaska	10	263	48
Arkansas	8.8	190	50
California	9	276	91
Florida	15.4	335	80
Georgia	17.4	167	60

USArests 14 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
29	13:12:45	select count(state) from USArests where Murder>8.1 LIMIT 0, 1000	1 row(s) returned	0.078 sec / 0.000 sec
30	13:13:45	select * from USArests where Murder>8.1 LIMIT 0, 1000	21 row(s) returned	0.093 sec / 0.000 sec

Object Info Session

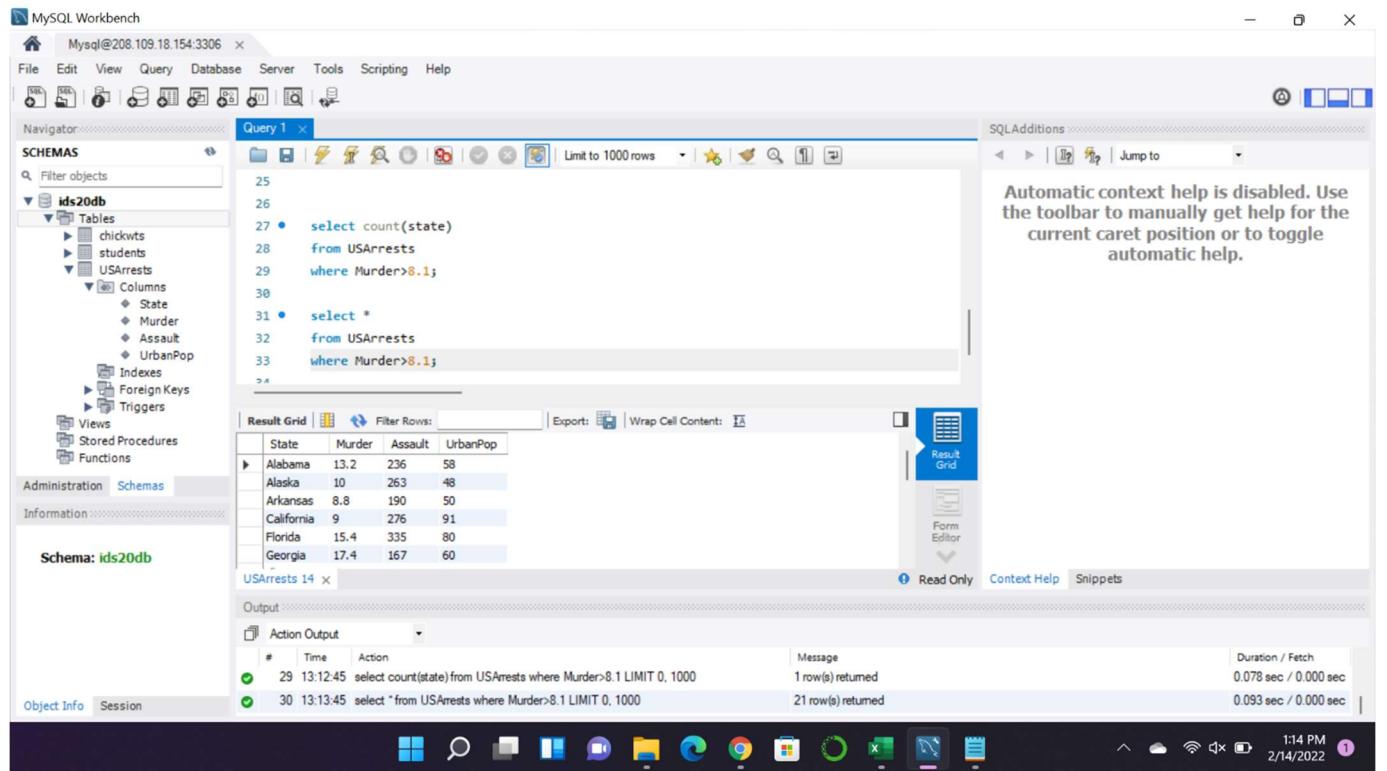
SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid | Form Editor

Read Only Context Help Snippets

1:14 PM 2/14/2022



Problem 2:

Excel:

Purpose of the Project:

To prepare the dataset for analysis by using the pre-processing Techniques.

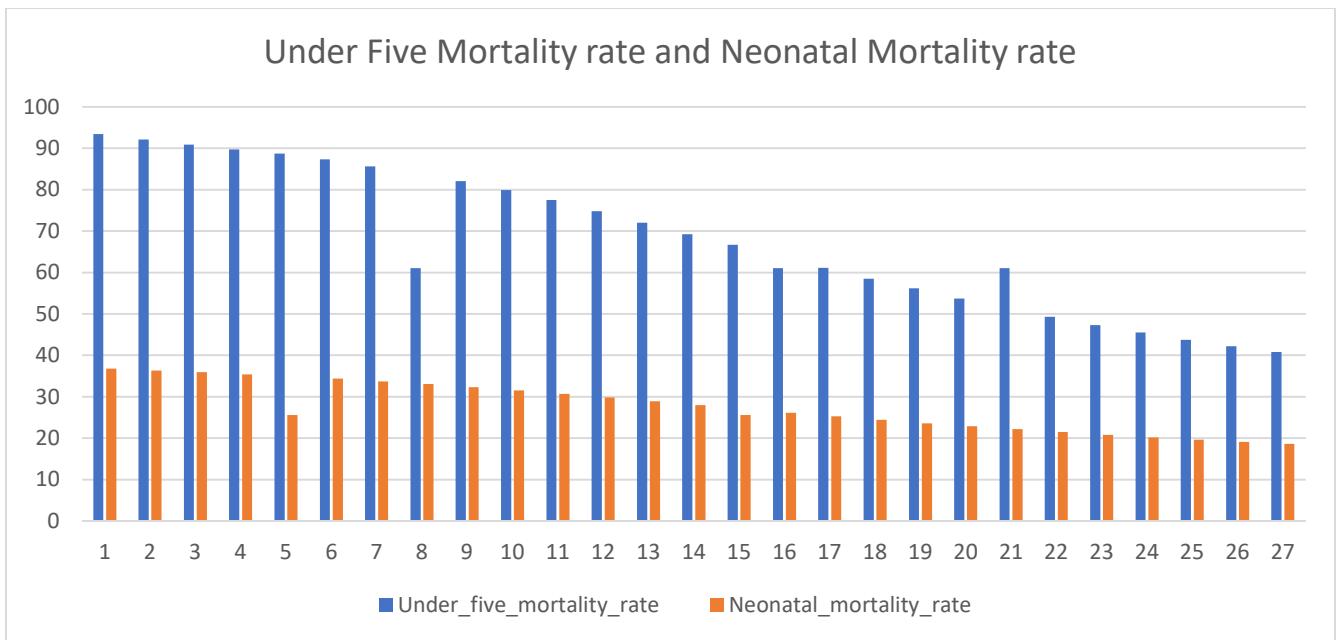
Operations:

- First took the Child mortality rate file in the professor Github Repository <https://github.com/bforoura/IDS/tree/main/HW2>.
- Imported the data into the Excel worksheet but I found several missing values in the data, and I applied the average method for every respective column to get the Mean values and I placed those values in place of the missing value for their respective columns.

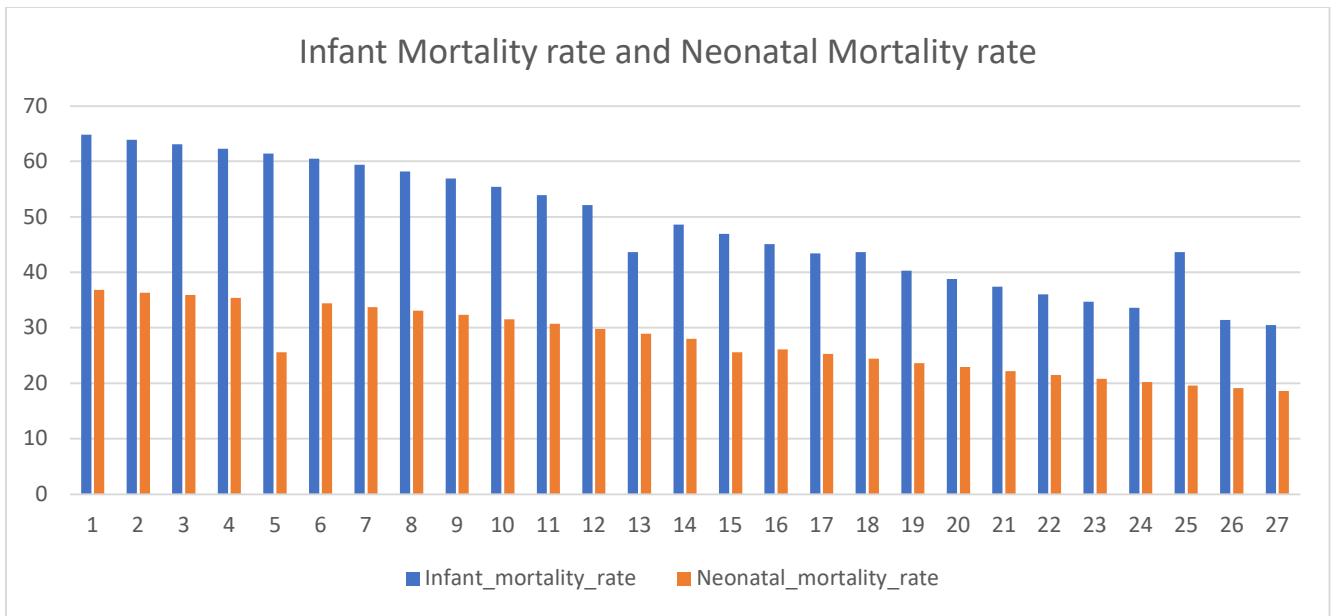
Below are the result after updating the missing values of the data

H10	A	B	C	D	E
1	State	Murder	Assault	UrbanPop	
2	Alabama	13.2	236	58	
3	Alaska	10	263	48	
4	Arizona	8.1	294	80	
5	Arkansas	8.8	190	50	
6	California	9	276	91	
7	Colorado	7.9	204	78	
8	Connecticut	3.3	110	77	
9	Delaware	5.9	238	72	
10	Florida	15.4	335	80	
11	Georgia	17.4	166	60	
12	Hawaii	5.3	46	83	
13	Idaho	2.6	120	54	
14	Illinois	10.4	249	83	
15	Indiana	7.2	113	65	
16	Iowa	2.2	56	57	
17	Kansas	6	115	66	
18	Kentucky	9.7	109	52	
19	Louisiana	15.4	249	66	
20	Maine	2.1	83	51	

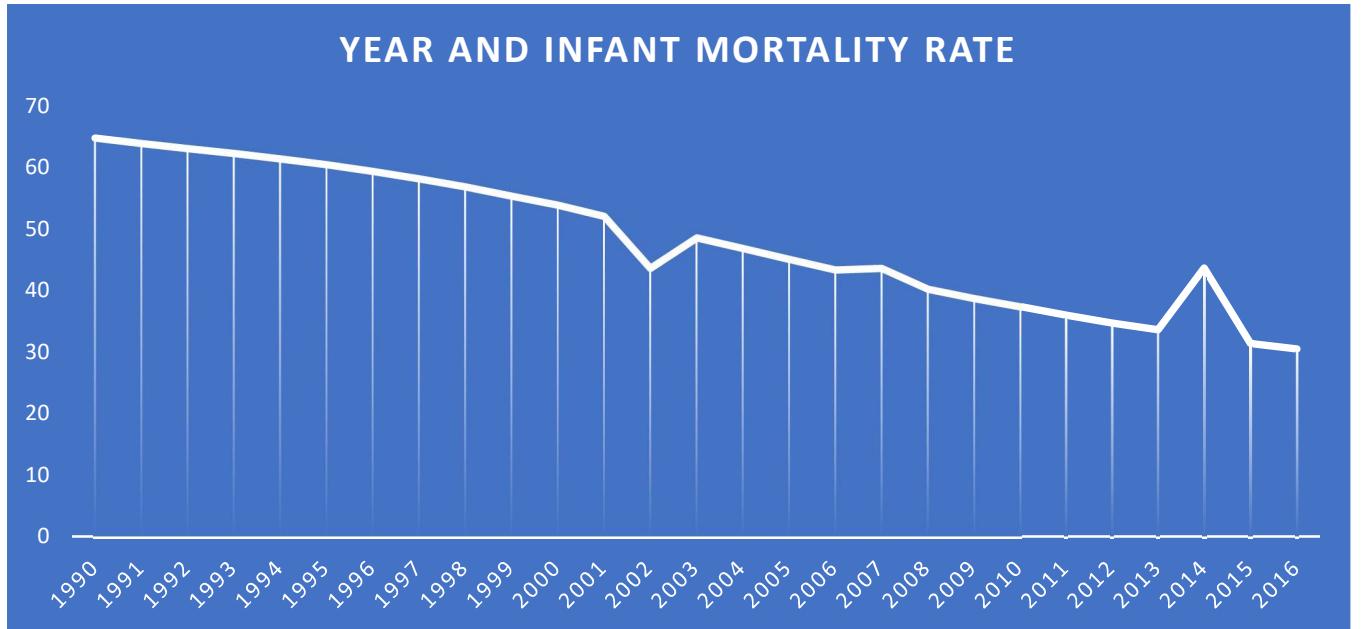
B. Under-five mortality rate and neonatal mortality rate.



Infant mortality rate and neonatal mortality rate.



Year and infant mortality rate.



Categoric relation between every attribute in the data.

CHILD MORTALITY NARESH

Naresh Kalluri NK

Comments Share

File Home Insert Draw Page Layout Formulas Data Review View Help

Font Alignment Number Styles Cells Editing Analysis

Format Shape

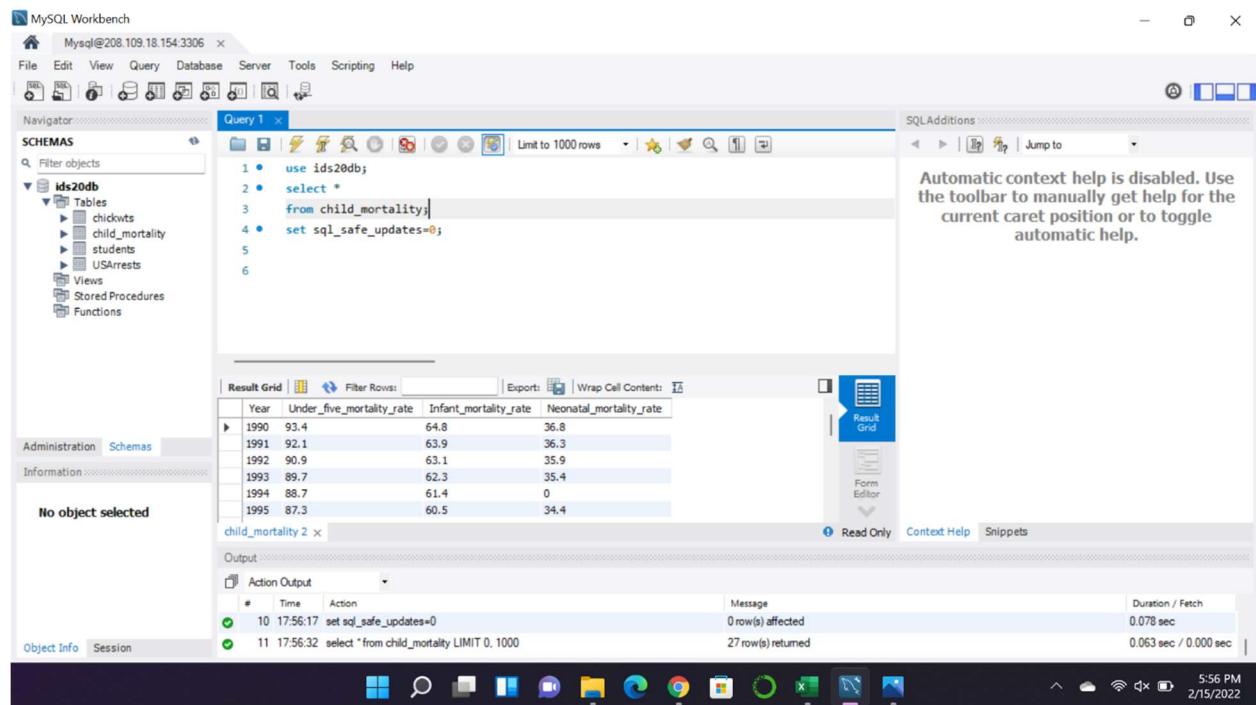
Fill Line

A	B	C	D	E	F	G	H	I	J	K
1	Year	Under_five_mortality_rate	Reation Category	Infant_mortality_rate	Infant category	Neonatal_mortality_rate	neonatal Category			
2	1990	93.4	VERY HIGH	64.8	HIGH	36.8	MEDIUM			
3	1991	92.1	VERY HIGH	63.9	HIGH	36.3	MEDIUM			
4	1992	90.9	VERY HIGH	63.1	HIGH	35.9	MEDIUM			
5	1993	89.7	HIGH	62.3	HIGH	35.4	MEDIUM			
6	1994	88.7	HIGH	61.4	HIGH	25.59	SMALL			
7	1995	87.3	HIGH	60.5	HIGH	34.4	MEDIUM			
8	1996	85.6	HIGH	59.4	MEDIUM	33.7	MEDIUM			
9	1997	61.04	HIGH	58.2	MEDIUM	33.1	MEDIUM			
10	1998	82.1	HIGH	56.9	MEDIUM	32.3	MEDIUM			
11	1999	79.9	HIGH	55.4	MEDIUM	31.5	MEDIUM			
12	2000	77.5	HIGH	53.9	MEDIUM	30.7	MEDIUM			
13	2001	74.8	HIGH	52.1	MEDIUM	29.8	SMALL			
14	2002	72	HIGH	43.65	MEDIUM	28.9	SMALL			
15	2003	69.2	HIGH	48.6	MEDIUM	28	SMALL			
16	2004	66.7	HIGH	46.9	MEDIUM	25.59	SMALL			
17	2005	61.04	HIGH	45.1	MEDIUM	26.1	SMALL			
18	2006	61.1	HIGH	43.4	MEDIUM	25.3	SMALL			
19	2007	58.5	MEDIUM	43.65	MEDIUM	24.4	SMALL			
20	2008	56.2	MEDIUM	40.3	MEDIUM	23.6	SMALL			
21	2009	53.7	MEDIUM	38.8	MEDIUM	22.9	SMALL			
22	2010	61.04	HIGH	37.4	MEDIUM	22.2	SMALL			
23	2011	49.3	MEDIUM	36	MEDIUM	21.5	SMALL			
24	2012	47.3	MEDIUM	34.7	MEDIUM	20.8	SMALL			
25	2013	45.5	MEDIUM	33.6	MEDIUM	20.2	SMALL			
26	2014	43.7	MEDIUM	43.65	MEDIUM	19.6	SMALL			

## MySQL:

### A. Import the original data set into MySQL:

Result: Imported the original data set into the MySQL workspace successfully.



The screenshot shows the MySQL Workbench interface. In the Navigator pane, the schema 'ids20db' is selected, displaying tables like 'chickwts', 'child\_mortality', 'students', and 'USArests'. The 'Query 1' editor contains the following SQL code:

```
1 • use ids20db;
2 • select *
3   from child_mortality;
4 • set sql_safe_updates=0;
5
6
```

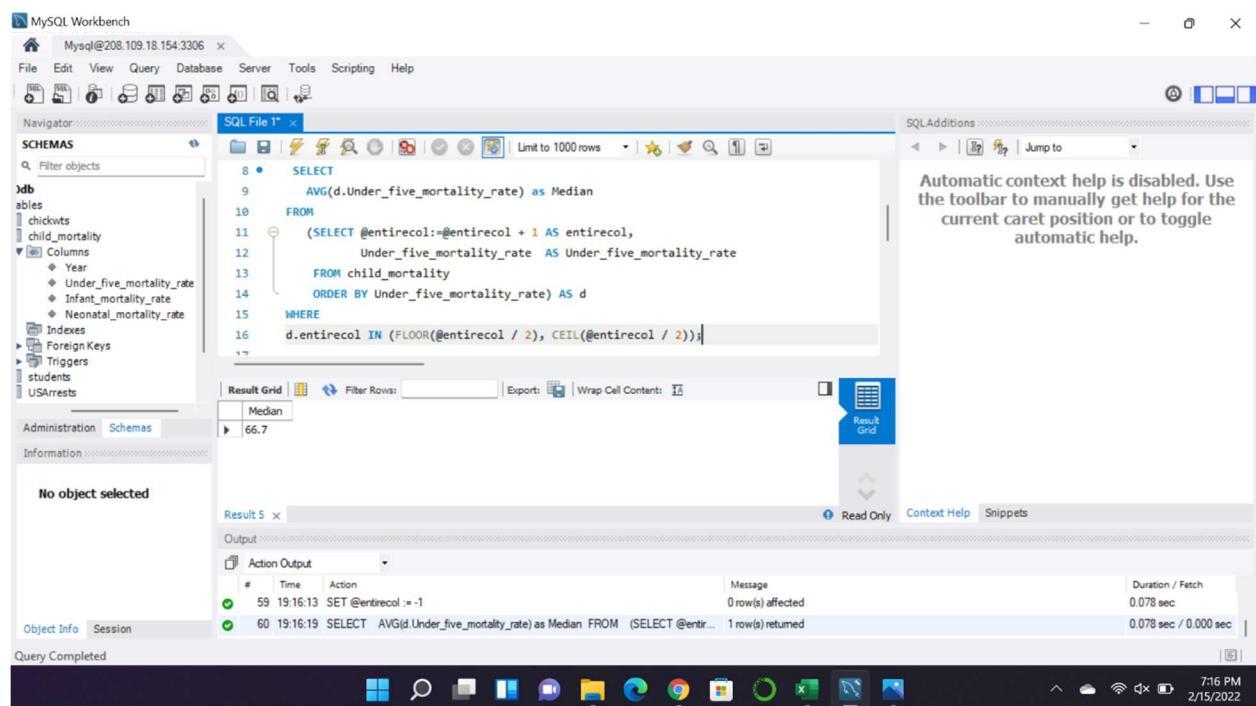
The Result Grid shows the data from the 'child\_mortality' table:

Year	Under_five_mortality_rate	Infant_mortality_rate	Neonatal_mortality_rate
1990	93.4	64.8	36.8
1991	92.1	63.9	36.3
1992	90.9	63.1	35.9
1993	89.7	62.3	35.4
1994	88.7	61.4	0
1995	87.3	60.5	34.4

The Output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
10	17:56:17	set sql_safe_updates=0	0 row(s) affected	0.078 sec
11	17:56:32	select * from child_mortality LIMIT 0, 1000	27 row(s) returned	0.063 sec / 0.000 sec

### B. Use SQL to fill in the missing values in each column using the medians.



The screenshot shows the MySQL Workbench interface. In the Navigator pane, the schema 'ids20db' is selected, displaying tables like 'chickwts', 'child\_mortality', 'students', and 'USArests'. The 'SQL File 1\*' editor contains the following SQL code:

```
8 • SELECT
9   AVG(d.Under_five_mortality_rate) AS Median
10  FROM
11    (SELECT @entirecol:=@entirecol + 1 AS entirecol,
12      Under_five_mortality_rate AS Under_five_mortality_rate
13     FROM child_mortality
14    ORDER BY Under_five_mortality_rate) AS d
15  WHERE
16    d.entirecol IN (FLOOR(@entirecol / 2), CEIL(@entirecol / 2));
17
```

The Result Grid shows the calculated median value:

Median
66.7

The Output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
59	19:16:13	SET @entirecol := 1	0 row(s) affected	0.078 sec
60	19:16:19	SELECT AVG(d.Under_five_mortality_rate) as Median FROM (SELECT @entir...	1 row(s) returned	0.078 sec / 0.000 sec

MySQL Workbench

Mysql@208.109.18.154:3306

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

db

Tables

chickwts

child\_mortality

Columns

Year

Under\_five\_mortality\_rate

Infant\_mortality\_rate

Neonatal\_mortality\_rate

Indexes

Foreign Keys

Triggers

students

USAArrests

Administration Schemas

Information

No object selected

Object Info Session

Query Completed

SQL File 1\*

```
20 •  SELECT
21      AVG(d.Infant_mortality_rate) as Median
22  FROM
23      (SELECT @entirecol:=@entirecol + 1 AS entirecol,
24          Infant_mortality_rate AS Infant_mortality_rate
25      FROM child_mortality
26      ORDER BY Infant_mortality_rate) AS d
27  WHERE
28      d.entirecol IN (FLOOR(@entirecol / 2), CEIL(@entirecol / 2))
```

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

Median  
46.9

Result 6 x

Output

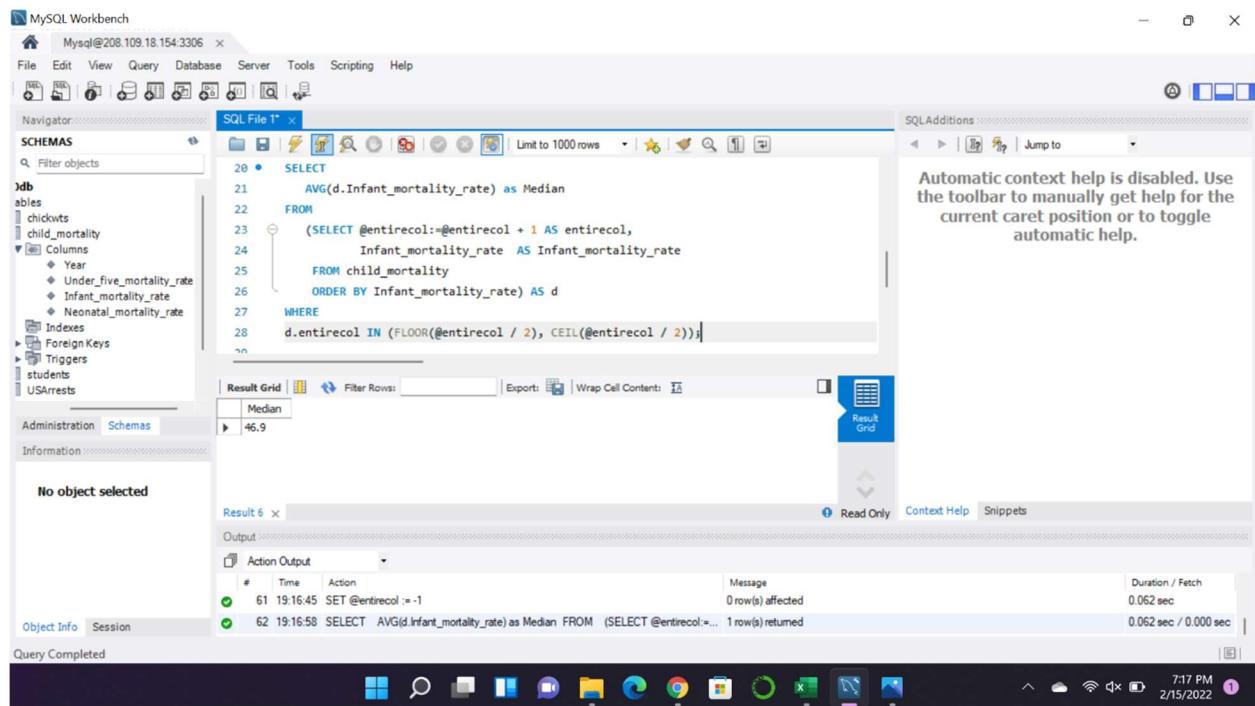
Action Output

#	Time	Action	Message	Duration / Fetch
61	19:16:45	SET @entirecol := -1	0 row(s) affected	0.062 sec
62	19:16:58	SELECT AVG(d.Infant_mortality_rate) as Median FROM (SELECT @entirecol =...	1 row(s) returned	0.062 sec / 0.000 sec

Read Only Context Help Snippets

7:17 PM 2/15/2022

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.



MySQL Workbench

Mysql@208.109.18.154:3306

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

db

Tables

chickwts

child\_mortality

Columns

Year

Under\_five\_mortality\_rate

Infant\_mortality\_rate

Neonatal\_mortality\_rate

Indexes

Foreign Keys

Triggers

students

USAArrests

Administration Schemas

Information

No object selected

Object Info Session

Query Completed

SQL File 1\*

```
32 •  SELECT
33      AVG(d.Neonatal_mortality_rate) as Median
34  FROM
35      (SELECT @entirecol:=@entirecol + 1 AS entirecol,
36          Neonatal_mortality_rate AS Neonatal_mortality_rate
37      FROM child_mortality
38      ORDER BY Neonatal_mortality_rate) AS d
39  WHERE
40      d.entirecol IN (FLOOR(@entirecol / 2), CEIL(@entirecol / 2))
```

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

Median  
26.1

Result 7 x

Output

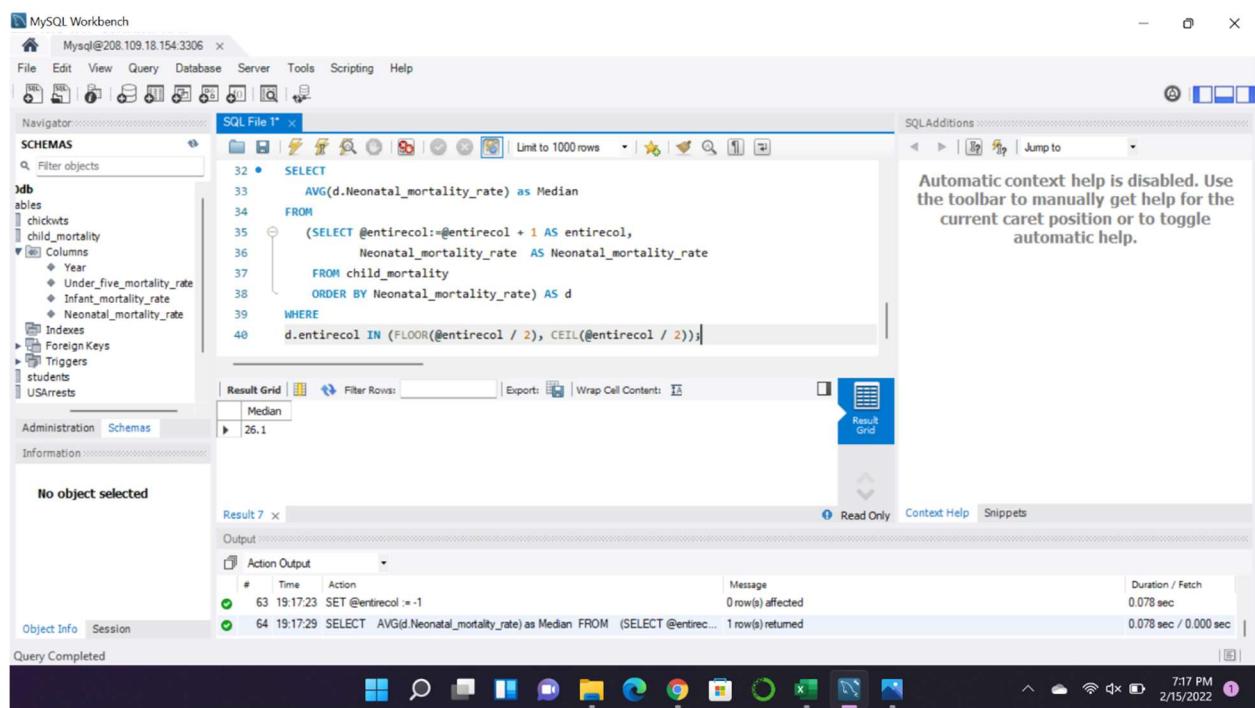
Action Output

#	Time	Action	Message	Duration / Fetch
63	19:17:23	SET @entirecol := -1	0 row(s) affected	0.078 sec
64	19:17:29	SELECT AVG(d.Neonatal_mortality_rate) as Median FROM (SELECT @entirec...	1 row(s) returned	0.078 sec / 0.000 sec

Read Only Context Help Snippets

7:17 PM 2/15/2022

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.



Display the entire table ;

The screenshot shows the MySQL Workbench interface. In the SQL Editor tab, the following SQL code is written:

```
49
50 • update child_mortality set neonatal_mortality_rate=26.1
51 where neonatal_mortality_rate=0;
52
53 • select *
54 from child_mortality;
55
56 • select max(infant_mortality_rate),min(infant_mortality_rate) from child_mortality;
57 • select year
58 from child_mortality;
```

The Result Grid shows the data from the child\_mortality table:

Year	Under_five_mortality_rate	Infant_mortality_rate	Neonatal_mortality_rate
2003	69.2	48.6	28
2004	66.7	46.9	26.1
2005	66.7	45.1	26.1
2006	61.1	43.4	25.3
2007	58.5	46.9	24.4

The Output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
70	19:23:55	select * from child_mortality LIMIT 0, 1000	27 row(s) returned	0.078 sec / 0.000 sec
71	19:24:30	select * from child_mortality LIMIT 0, 1000	27 row(s) returned	0.078 sec / 0.000 sec

Which years have the lowest and highest infant mortality years, respectively?

The screenshot shows the MySQL Workbench interface. In the SQL Editor tab, the following SQL code is written:

```
49
50 • update child_mortality set neonatal_mortality_rate=26.1
51 where neonatal_mortality_rate=0;
52
53 • select *
54 from child_mortality;
55
56 • select max(infant_mortality_rate),min(infant_mortality_rate) from child_mortality;
57 • select year
58 from child_mortality;
```

The Result Grid shows the results of the MAX and MIN functions:

max(infant_mortality_rate)	min(infant_mortality_rate)
64.8	30.5

The Output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
71	19:24:30	select * from child_mortality LIMIT 0, 1000	27 row(s) returned	0.078 sec / 0.000 sec
72	19:24:58	select max(infant_mortality_rate),min(infant_mortality_rate) from child_mortality LIMIT... 1 row(s) returned		0.078 sec / 0.000 sec

In what years the neonatal mortality rates were above average?

The screenshot shows the MySQL Workbench interface. In the top-left, the connection details are shown as "Mysql@208.109.18.154:3306". The main area displays a SQL editor window titled "CHILD MORTALITY NARESH\*". The query entered is:

```
58 • select *  
59   from child_mortality;  
60  
61 • select max(infant_mortality_rate),min(infant_mortality_rate) from child_mortality;  
62 • select year  
63   from child_mortality  
64   where infant_mortality_rate=64.8;  
65  
66 • select Year  
67   from child_mortality;
```

The result grid shows a single row with the value "1990" under the column "year". Below the editor, the status bar indicates "Query Completed".

At the bottom right, the Windows taskbar shows the date and time as "10:37 PM 2/15/2022".

The screenshot shows the MySQL Workbench interface. In the top-left, the connection details are shown as "Mysql@208.109.18.154:3306". The main area displays a SQL editor window titled "CHILD MORTALITY NARESH\*". The query entered is:

```
62 • select year  
63   from child_mortality  
64   where infant_mortality_rate=64.8;  
65  
66 • select Year  
67   from child_mortality  
68   where infant_mortality_rate=30.5;  
69  
70 • select avg(under_five_mortality_rate),avg(infant_mortality_rate),avg(neonatal_mortality)  
71   from child_mortality;
```

The result grid shows a single row with the value "2016" under the column "Year". Below the editor, the status bar indicates "Query Completed".

At the bottom right, the Windows taskbar shows the date and time as "10:37 PM 2/15/2022".

Display the sorted infant mortality rates in descending order.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the database schema with tables like `child\_mortality` and `USArrests`. The main query editor window contains the following SQL code:

```
1 from child_mortality;
2
3 • select Year
4   from child_mortality
5   where neonatal_mortality_rate >= 27.49;
6
7 • select infant_mortality_rate
8   from child_mortality
9   order by infant_mortality_rate desc;
```

The results grid shows the following data:

infant_mortality_rate
64.8
63.9
63.1
62.3
61.4
60.5

The status bar at the bottom right indicates "Query Completed" at 10:41 PM on 2/15/2022.

Display min, max, mean, variance, and standard deviation for each mortality rate.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the database schema. The main query editor window contains the following SQL code:

```
1 where neonatal_mortality_rate >= 27.49;
2
3 • select infant_mortality_rate
4   from child_mortality
5   order by infant_mortality_rate desc;
6
7 • select min(under_five_mortality_rate) as under_min,max(under_five_mortality_rate) as under_max,min(under_five_mortality_rate) as under_avg,avg(under_five_mortality_rate) as under_var,min(neonatal_mortality_rate) as neo_min,max(neonatal_mortality_rate) as neo_max,avg(neonatal_mortality_rate) as neo_avg,avg(neonatal_mortality_rate) as neo_var
8
```

The results grid shows the following data:

under_min	under_max	under_avg	under_var	inf_min	inf_max	inf_avg	inf_var
40.8	93.4	68.45555555555555	281.0928395061728	30.5	64.8	48.86296296296296	114.9615912

The status bar at the bottom right indicates "Query Completed" at 10:44 PM on 2/15/2022.

Add a new column called **Above-Five Mortality Rate** and populate it with appropriate values.

The screenshot shows the MySQL Workbench interface. In the SQL Editor tab, the following SQL code is run:

```
78 • desc child_mortality;
80 • alter table child_mortality add column Above_Five_Mortality_Rate double;
82
83 • desc child_mortality;
84
85 • update child_mortality set Above_Five_Mortality_Rate=under_five_mortality_rate+5;
86 • select * from child_mortality;
```

The Result Grid displays the data from the child\_mortality table, including the newly added column:

Year	Under_five_mortality_rate	Infant_mortality_rate	Neonatal_mortality_rate	Above_Five_Mortality_Rate
1990	93.4	64.8	36.8	98.4
1991	92.1	63.9	36.3	97.1
1992	90.9	63.1	35.9	95.9
1993	89.7	62.3	35.4	94.7
1994	88.7	61.4	26.1	93.7
...	...	...	...	...

The Output tab shows the execution details:

#	Time	Action	Message	Duration / Fetch
84	19:31:34	select min(under_five_mortality_rate) as under_min,max(under_five_mortality_rate) as under_max from child_mortality LIMIT 0 , 1000	1 row(s) returned	0.078 sec / 0.000 sec
85	19:31:36	update child_mortality set Above_Five_Mortality_Rate=under_five_mortality_rate+5;	27 rows affected Rows matched: 27 Changed: 0 Warnings: 0	0.078 sec / 0.000 sec

At the bottom right, the system tray shows the date and time as 2/15/2022 7:33 PM.

Display the entire table again.

The screenshot shows the MySQL Workbench interface. In the SQL Editor tab, the following SQL code is run:

```
84
85 • desc child_mortality;
86
87 • alter table child_mortality add column Above_Five_Mortality_Rate double;
88
89 • update child_mortality set Above_Five_Mortality_Rate=under_five_mortality_rate+5;
90
91 • select *
92 • from child_mortality;
```

The Result Grid displays the data from the child\_mortality table, including the newly added column:

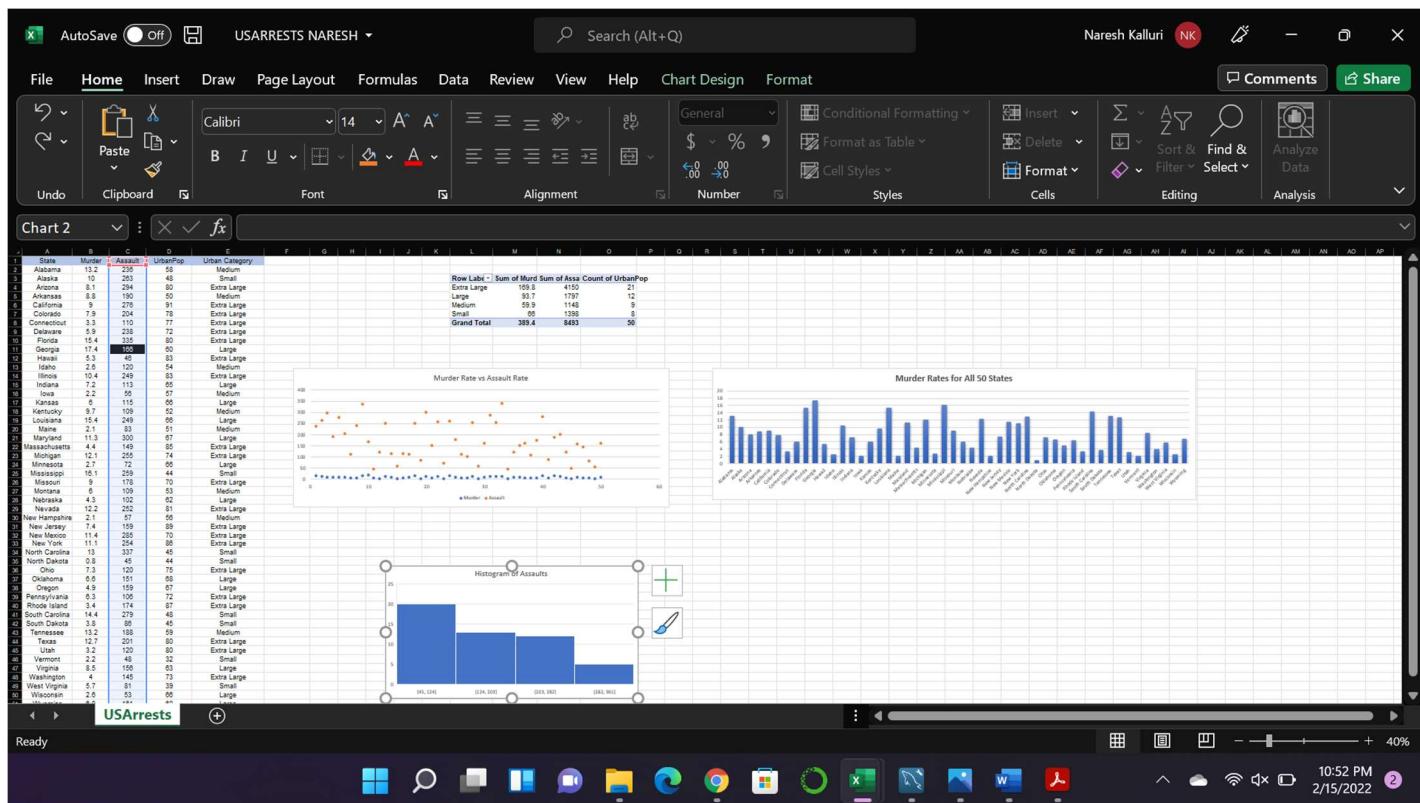
Year	Under_five_mortality_rate	Infant_mortality_rate	Neonatal_mortality_rate	Above_Five_Mortality_Rate
1990	93.4	64.8	36.8	98.4
1991	92.1	63.9	36.3	97.1
1992	90.9	63.1	35.9	95.9
1993	89.7	62.3	35.4	94.7
1994	88.7	61.4	26.1	93.7
...	...	...	...	...

The Output tab shows the execution details:

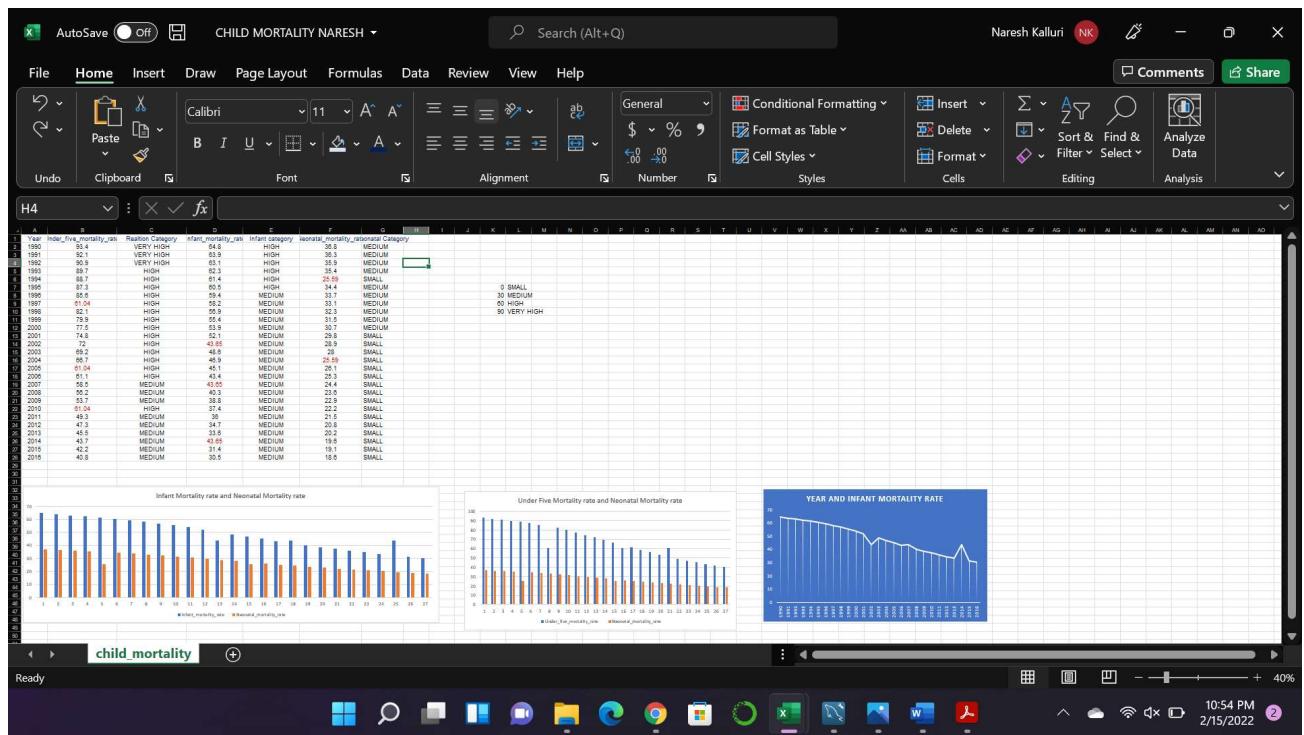
#	Time	Action	Message	Duration / Fetch
108	22:47:11	update child_mortality set Above_Five_Mortality_Rate=under_five_mortality_rate+5;	0 row(s) affected Rows matched: 27 Changed: 0 Warnings: 0	0.078 sec
109	22:47:14	select * from child_mortality LIMIT 0 , 1000	27 rows affected Rows matched: 27 Changed: 0 Warnings: 0	0.078 sec

At the bottom right, the system tray shows the date and time as 2/15/2022 10:47 PM.

## Excel Worksheets: Problem 1:



## Problem 2:



References:

<https://www.geeksforgeeks.org/calculate-median-in-mysql/>

Used for how to calculate Median in MySQL

I converted my cleansed data into XML and JSON formats and uploaded it to my GitHub repository including my Excel Worksheets.