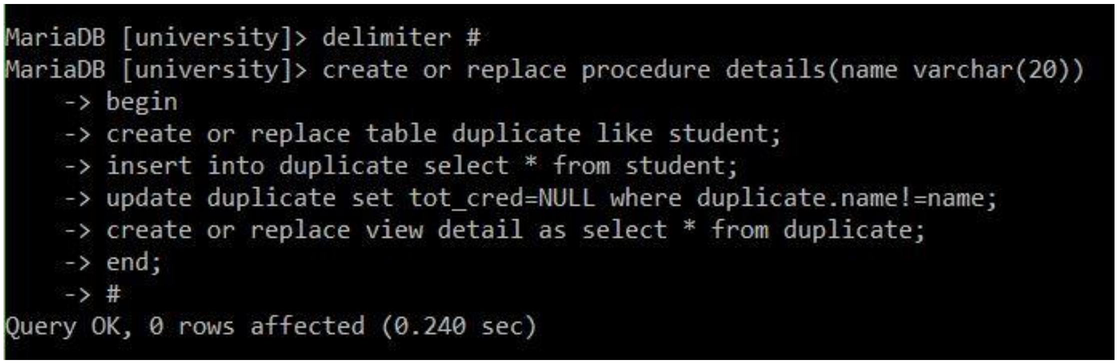
ASSIGNMENT – 7 ABHICHAL RAJPUT

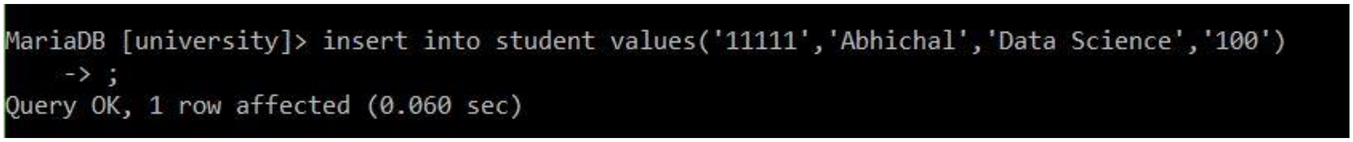
1. Create a procedure which will be used by a system admin wheneven a new student joins. The procedure will create a view such that, each student can see details of all students except their total credit, but the student can see all information about him/her. Demonstrate this procedure with examples. [3]

Step1:- Procedure for creating the view which is called by system admin



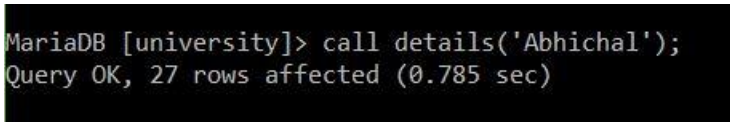
```
MariaDB [university]> delimiter #
MariaDB [university]> create or replace procedure details(name varchar(20))
    -> begin
    -> create or replace table duplicate like student;
    -> insert into duplicate select * from student;
    -> update duplicate set tot_cred=NULL where duplicate.name!=name;
    -> create or replace view detail as select * from duplicate;
    -> end;
    -> #
Query OK, 0 rows affected (0.240 sec)
```

Step2:- Inserting details for new student into the student table



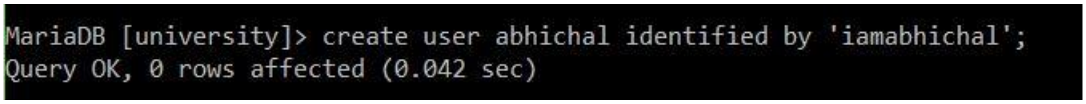
```
MariaDB [university]> insert into student values('11111','Abhichal','Data Science','100')
    -> ;
Query OK, 1 row affected (0.060 sec)
```

Step3:- System admin calling the procedure which is created above



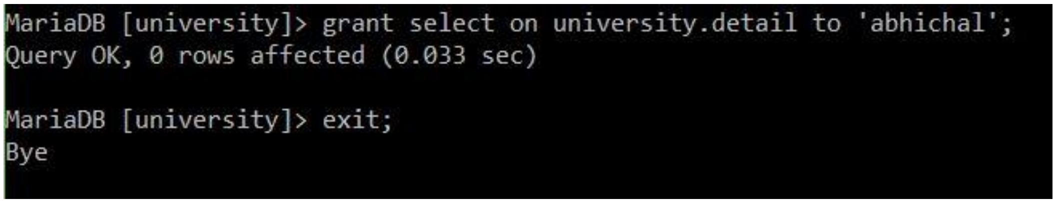
```
MariaDB [university]> call details('Abhichal');
Query OK, 27 rows affected (0.785 sec)
```

Step4:- Creating a user



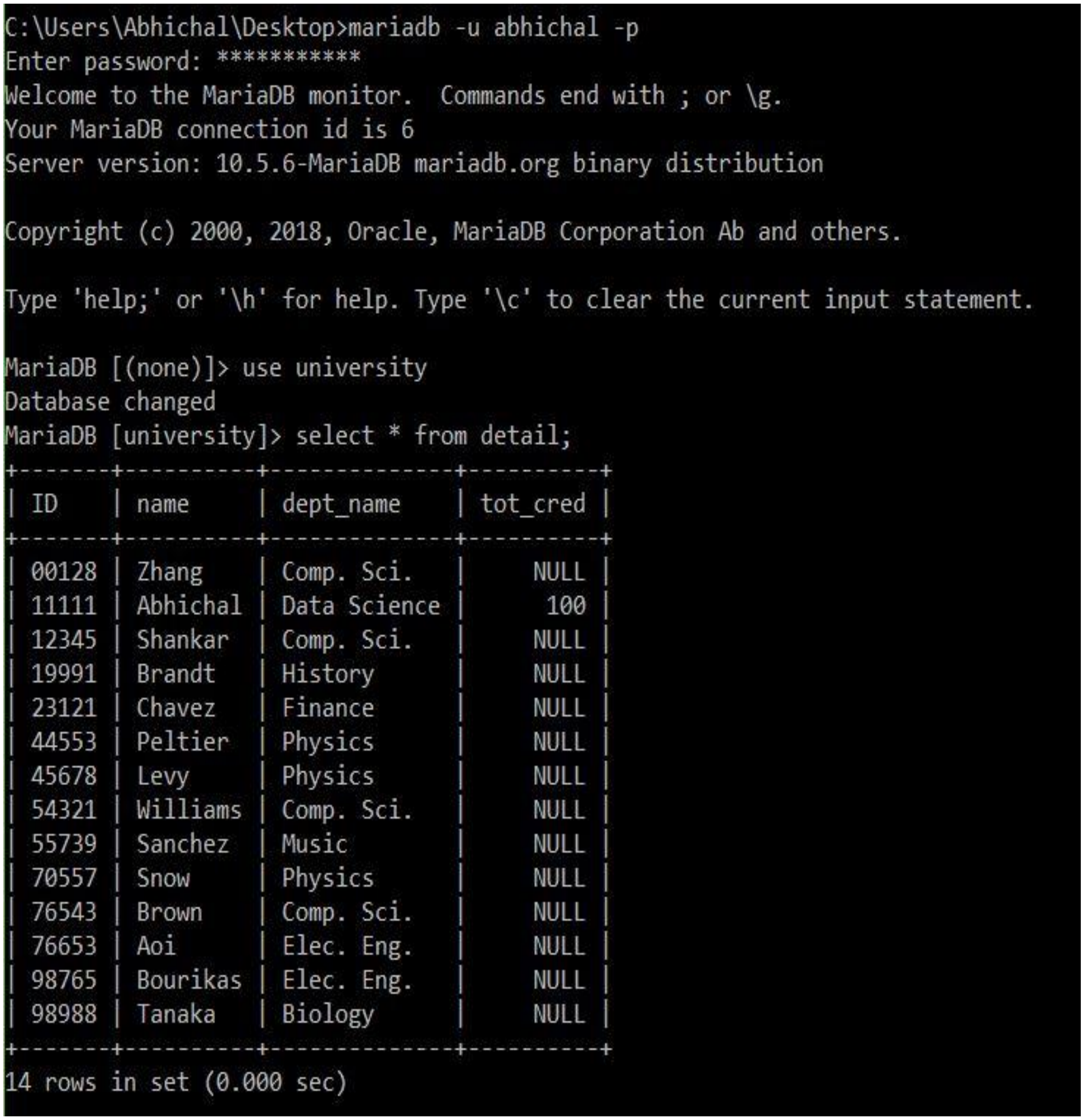
```
MariaDB [university]> create user abhichal identified by 'iamabhichal';
Query OK, 0 rows affected (0.042 sec)
```

Step5:- Granting permission to the user and exit from database



```
MariaDB [university]> grant select on university.detail to 'abhichal';
Query OK, 0 rows affected (0.033 sec)

MariaDB [university]> exit;
Bye
```

Step6:- Entering the database as a user and accessing the view.

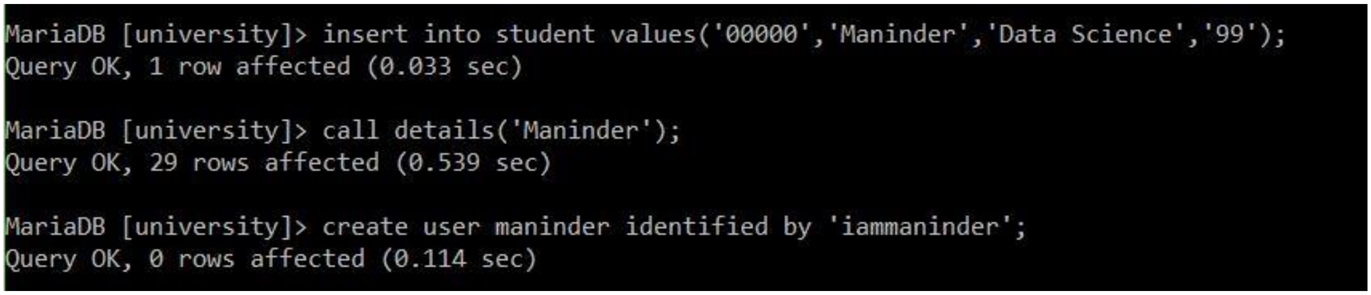


```
C:\Users\Abhichal\Desktop>mariadb -u abhichal -p
Enter password: ***********
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 6
Server version: 10.5.6-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use university
Database changed
MariaDB [university]> select * from detail;
+-------+----------+--------------+----------+
| ID    | name     | dept_name    | tot_cred |
+-------+----------+--------------+----------+
| 00128 | Zhang    | Comp. Sci.   |     NULL |
| 11111 | Abhichal | Data Science |      100 |
| 12345 | Shankar  | Comp. Sci.   |     NULL |
| 19991 | Brandt   | History      |     NULL |
| 23121 | Chavez   | Finance      |     NULL |
| 44553 | Peltier  | Physics      |     NULL |
| 45678 | Levy     | Physics      |     NULL |
| 54321 | Williams | Comp. Sci.   |     NULL |
| 55739 | Sanchez  | Music        |     NULL |
| 70557 | Snow     | Physics      |     NULL |
| 76543 | Brown    | Comp. Sci.   |     NULL |
| 76653 | Aoi      | Elec. Eng.   |     NULL |
| 98765 | Bourikas | Elec. Eng.   |     NULL |
| 98988 | Tanaka   | Biology      |     NULL |
+-------+----------+--------------+----------+
14 rows in set (0.000 sec)
```

ASSIGNMENT – 7 ABHICHAL RAJPUT

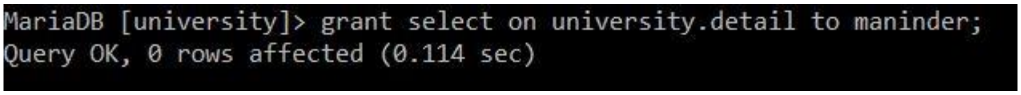
SAME PROCESS ON DIFFERENT USER:-

--Inserting the data then calling procedure and then creating a user



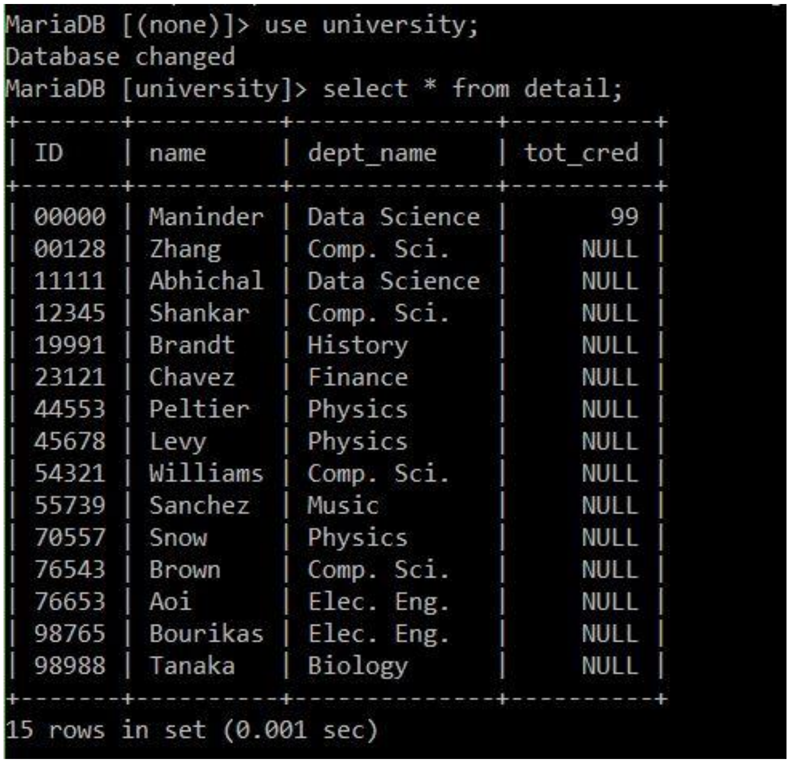
```
MariaDB [university]> insert into student values('00000','Maninder','Data Science','99');
Query OK, 1 row affected (0.033 sec)

MariaDB [university]> call details('Maninder');
Query OK, 29 rows affected (0.539 sec)

MariaDB [university]> create user maninder identified by 'iammaninder';
Query OK, 0 rows affected (0.114 sec)
```

--Granting permissions to the user



```
MariaDB [university]> grant select on university.detail to maninder;
Query OK, 0 rows affected (0.114 sec)
```

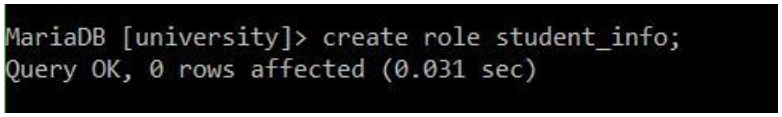
--Accessing as a user and seeing the view



```
MariaDB [(none)]> use university;
Database changed
MariaDB [university]> select * from detail;
+-------+----------+--------------+----------+
| ID    | name     | dept_name    | tot_cred |
+-------+----------+--------------+----------+
| 00000 | Maninder | Data Science |       99 |
| 00128 | Zhang    | Comp. Sci.   |     NULL |
| 11111 | Abhichal | Data Science |     NULL |
| 12345 | Shankar  | Comp. Sci.   |     NULL |
| 19991 | Brandt   | History      |     NULL |
| 23121 | Chavez   | Finance      |     NULL |
| 44553 | Peltier  | Physics      |     NULL |
| 45678 | Levy     | Physics      |     NULL |
| 54321 | Williams | Comp. Sci.   |     NULL |
| 55739 | Sanchez  | Music        |     NULL |
| 70557 | Snow     | Physics      |     NULL |
| 76543 | Brown    | Comp. Sci.   |     NULL |
| 76653 | Aoi      | Elec. Eng.   |     NULL |
| 98765 | Bourikas | Elec. Eng.   |     NULL |
| 98988 | Tanaka   | Biology      |     NULL |
+-------+----------+--------------+----------+
15 rows in set (0.001 sec)
```

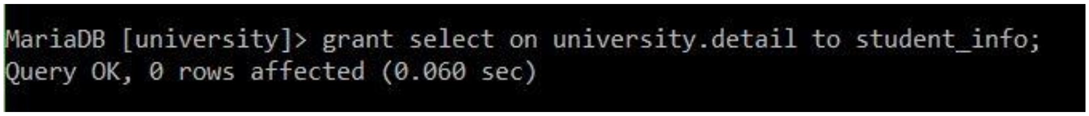
2. Can we solve the above task using role ? Justify your answer with examples using university database and executing commands. [1]

Step1:- Creating the role



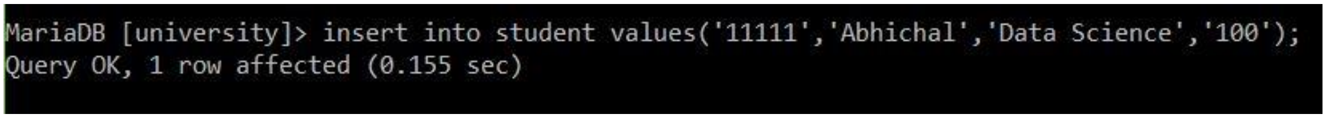
```
MariaDB [university]> create role student_info;
Query OK, 0 rows affected (0.031 sec)
```

Step2:- Granting permissions to the role



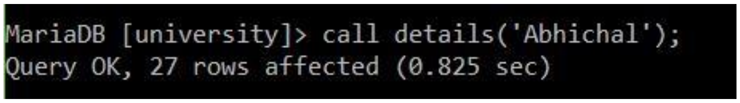
```
MariaDB [university]> grant select on university.detail to student_info;
Query OK, 0 rows affected (0.060 sec)
```

Step3:- Inserting the data of new student to the table



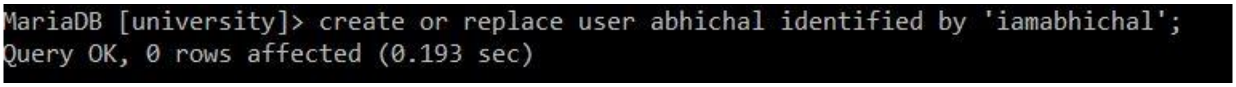
```
MariaDB [university]> insert into student values('11111','Abhichal','Data Science','100');
Query OK, 1 row affected (0.155 sec)
```

Step4:- calling the procedure



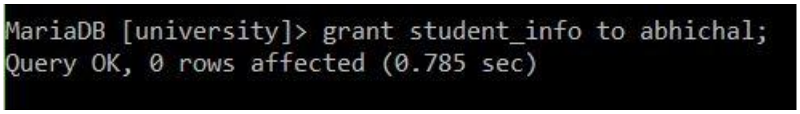
```
MariaDB [university]> call details('Abhichal');
Query OK, 27 rows affected (0.825 sec)
```

Step5:- create the user



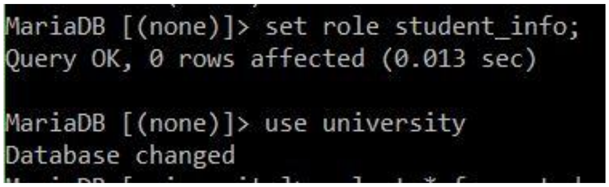
```
MariaDB [university]> create or replace user abhichal identified by 'iamabhichal';
Query OK, 0 rows affected (0.193 sec)
```

Step6:- set role to user and exit



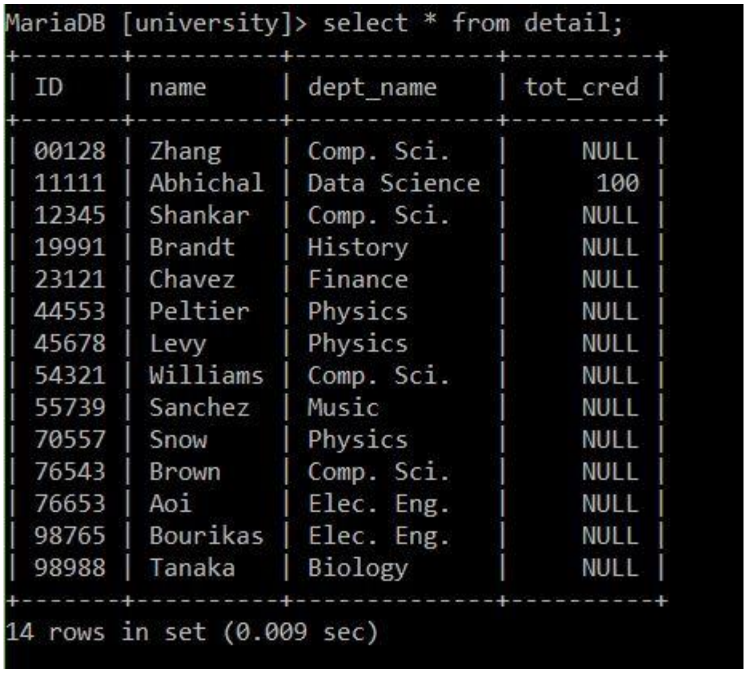
```
MariaDB [university]> grant student_info to abhichal;
Query OK, 0 rows affected (0.785 sec)
```

Step7:- Login through the user and accessing the database by setting the role



```
MariaDB [(none)]> set role student_info;
Query OK, 0 rows affected (0.013 sec)

MariaDB [(none)]> use university
Database changed
```

Step8:- Getting the details

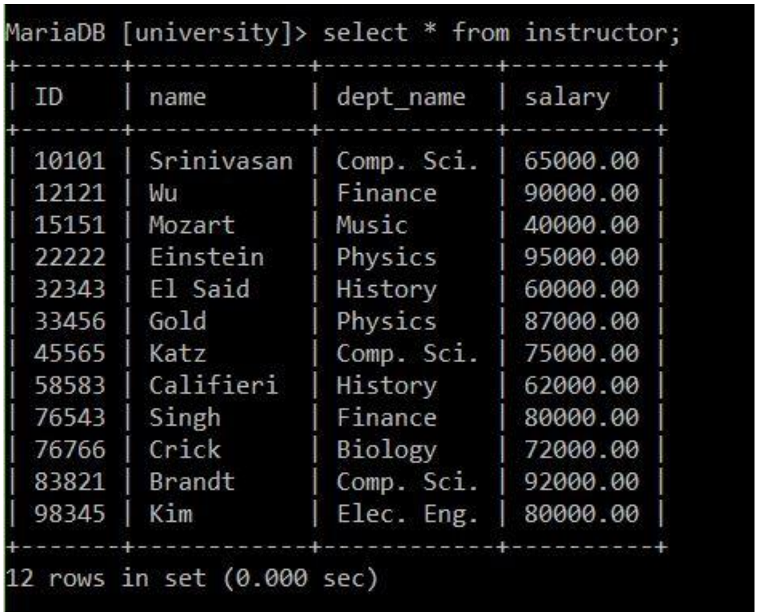


```
MariaDB [university]> select * from detail;
+-------+----------+--------------+----------+
| ID    | name     | dept_name    | tot_cred |
+-------+----------+--------------+----------+
| 00128 | Zhang    | Comp. Sci.   |     NULL |
| 11111 | Abhichal | Data Science |      100 |
| 12345 | Shankar  | Comp. Sci.   |     NULL |
| 19991 | Brandt   | History      |     NULL |
| 23121 | Chavez   | Finance      |     NULL |
| 44553 | Peltier  | Physics      |     NULL |
| 45678 | Levy     | Physics      |     NULL |
| 54321 | Williams | Comp. Sci.   |     NULL |
| 55739 | Sanchez  | Music        |     NULL |
| 70557 | Snow     | Physics      |     NULL |
| 76543 | Brown    | Comp. Sci.   |     NULL |
| 76653 | Aoi      | Elec. Eng.   |     NULL |
| 98765 | Bourikas | Elec. Eng.   |     NULL |
| 98988 | Tanaka   | Biology      |     NULL |
+-------+----------+--------------+----------+
14 rows in set (0.009 sec)
```

3. Demonstrate transaction using university database on instructor table. You can choose your case but explain it properly. [1]

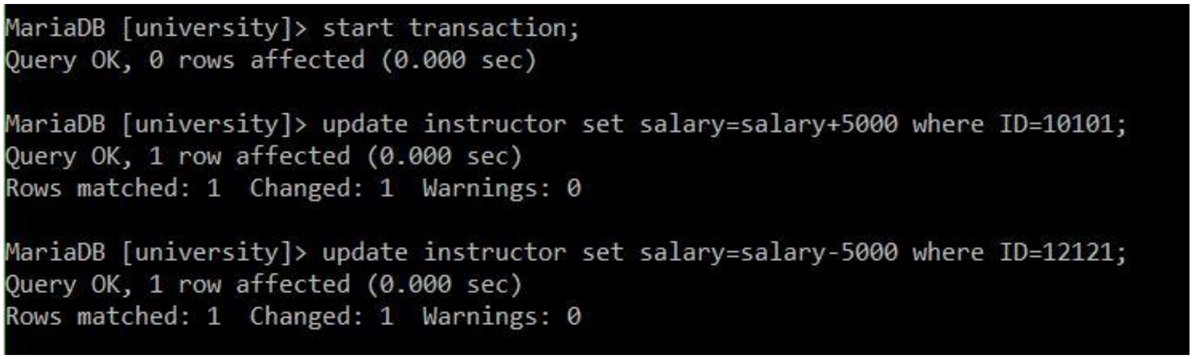
Solution :- Explaining the transaction with the help of instructor table.

Step 1:- Initial table before any operation.



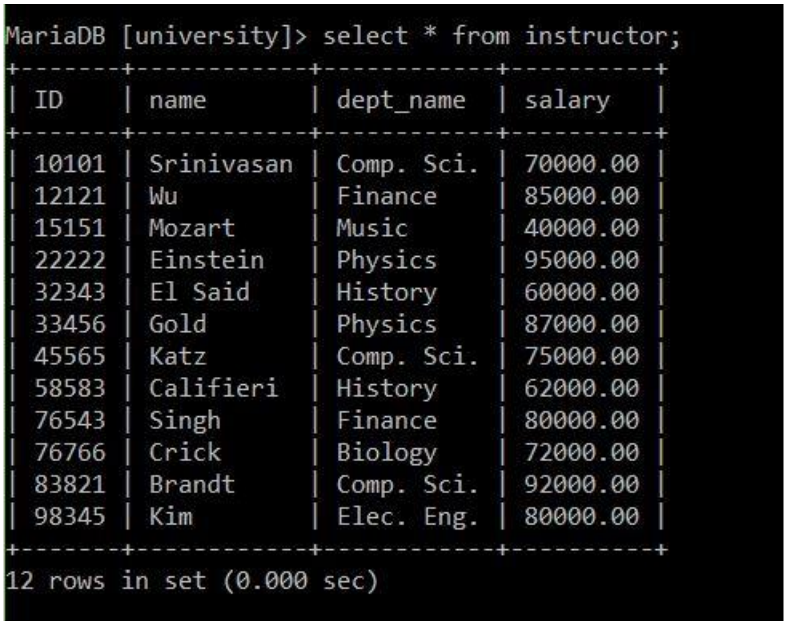
```
MariaDB [university]> select * from instructor;
+-------+------------+------------+----------+
| ID    | name       | dept_name  | salary   |
+-------+------------+------------+----------+
| 10101 | Srinivasan | Comp. Sci. | 65000.00 |
| 12121 | Wu         | Finance    | 90000.00 |
| 15151 | Mozart     | Music      | 40000.00 |
| 22222 | Einstein   | Physics    | 95000.00 |
| 32343 | El Said    | History    | 60000.00 |
| 33456 | Gold       | Physics    | 87000.00 |
| 45565 | Katz       | Comp. Sci. | 75000.00 |
| 58583 | Califieri  | History    | 62000.00 |
| 76543 | Singh      | Finance    | 80000.00 |
| 76766 | Crick      | Biology    | 72000.00 |
| 83821 | Brandt     | Comp. Sci. | 92000.00 |
| 98345 | Kim        | Elec. Eng. | 80000.00 |
+-------+------------+------------+----------+
12 rows in set (0.000 sec)
```

Step 2:- Starting a transaction and try to update the salary of Srinivasan by the same amount of salary Wu get a decrement. Let say 5000, so Srinivasan get an increment of 5000 in his salary while Wu get a decrement of 5000.



```
MariaDB [university]> start transaction;
Query OK, 0 rows affected (0.000 sec)

MariaDB [university]> update instructor set salary=salary+5000 where ID=10101;
Query OK, 1 row affected (0.000 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [university]> update instructor set salary=salary-5000 where ID=12121;
Query OK, 1 row affected (0.000 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

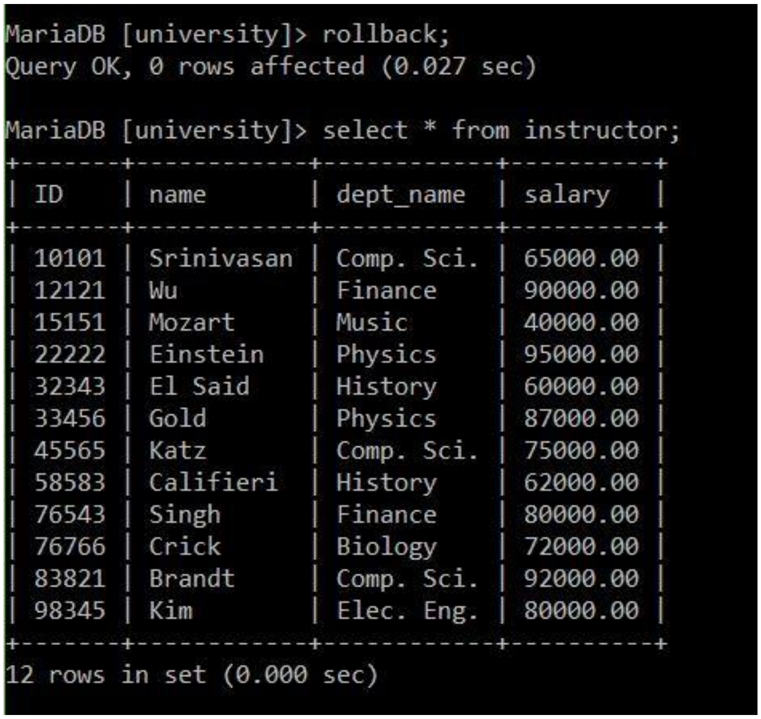
Step 3:- Now let’s check the updated table.



```
MariaDB [university]> select * from instructor;
+-------+------------+------------+----------+
| ID    | name       | dept_name  | salary   |
+-------+------------+------------+----------+
| 10101 | Srinivasan | Comp. Sci. | 70000.00 |
| 12121 | Wu         | Finance    | 85000.00 |
| 15151 | Mozart     | Music      | 40000.00 |
| 22222 | Einstein   | Physics    | 95000.00 |
| 32343 | El Said    | History    | 60000.00 |
| 33456 | Gold       | Physics    | 87000.00 |
| 45565 | Katz       | Comp. Sci. | 75000.00 |
| 58583 | Califieri  | History    | 62000.00 |
| 76543 | Singh      | Finance    | 80000.00 |
| 76766 | Crick      | Biology    | 72000.00 |
| 83821 | Brandt     | Comp. Sci. | 92000.00 |
| 98345 | Kim        | Elec. Eng. | 80000.00 |
+-------+------------+------------+----------+
12 rows in set (0.000 sec)
```

We can see the salary of Srinivasan increased by 5000 and Wu decreased by 5000.

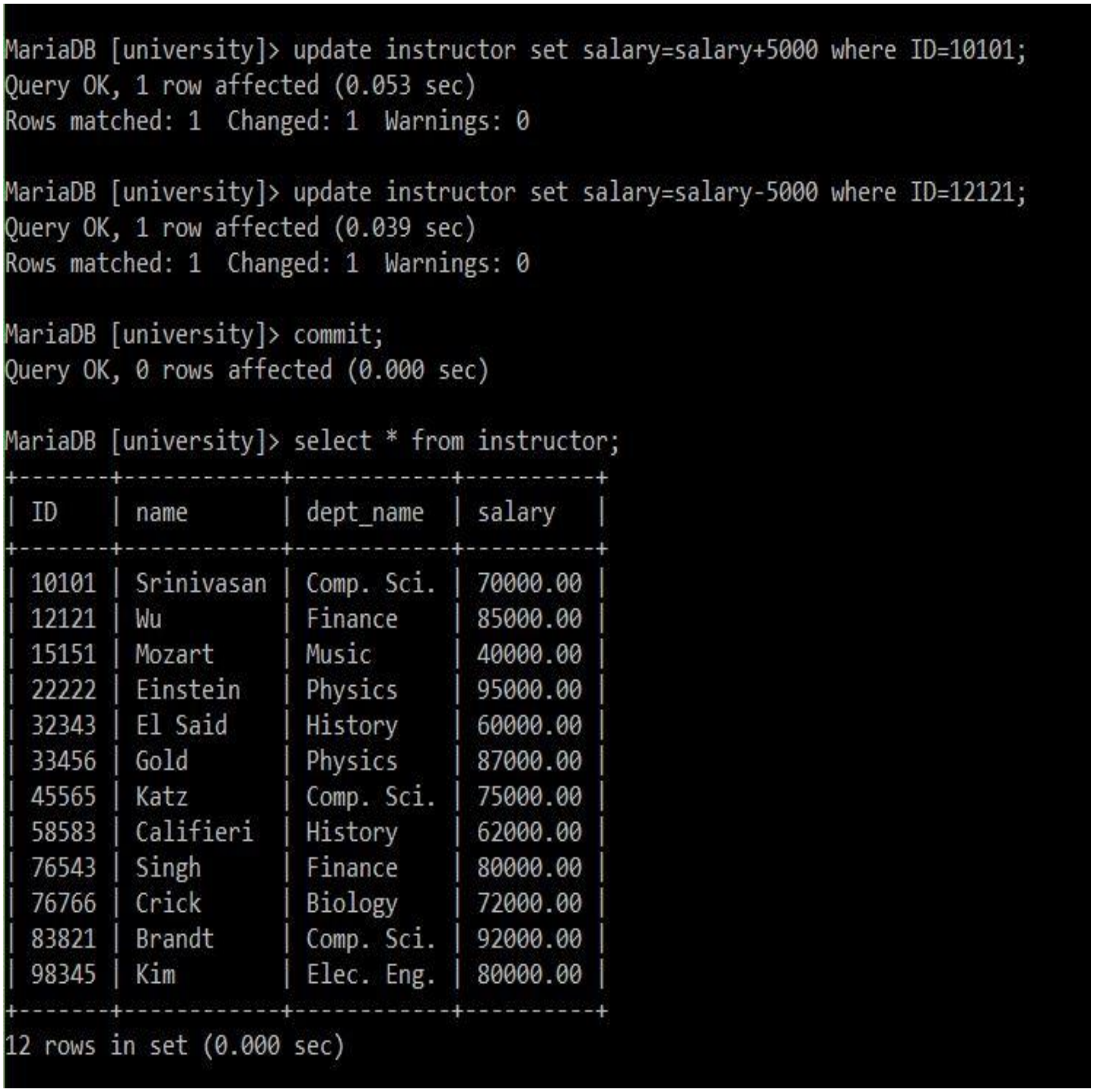
Step 4:- Now if, this is not the desired amount we have to update we can reach the previous state by doing rollback. Let’s see….



```
MariaDB [university]> rollback;
Query OK, 0 rows affected (0.027 sec)

MariaDB [university]> select * from instructor;
+-------+------------+------------+----------+
| ID    | name       | dept_name  | salary   |
+-------+------------+------------+----------+
| 10101 | Srinivasan | Comp. Sci. | 65000.00 |
| 12121 | Wu         | Finance    | 90000.00 |
| 15151 | Mozart     | Music      | 40000.00 |
| 22222 | Einstein   | Physics    | 95000.00 |
| 32343 | El Said    | History    | 60000.00 |
| 33456 | Gold       | Physics    | 87000.00 |
| 45565 | Katz       | Comp. Sci. | 75000.00 |
| 58583 | Califieri  | History    | 62000.00 |
| 76543 | Singh      | Finance    | 80000.00 |
| 76766 | Crick      | Biology    | 72000.00 |
| 83821 | Brandt     | Comp. Sci. | 92000.00 |
| 98345 | Kim        | Elec. Eng. | 80000.00 |
+-------+------------+------------+----------+
12 rows in set (0.000 sec)
```

We can see that after performing rollback the Salaries of Srinivasan and Wu are setted to initial salary. This is because we didn’t commit before the rollback means we didn’t save the state using commit. So, rollback sent the state to the previous committed state.

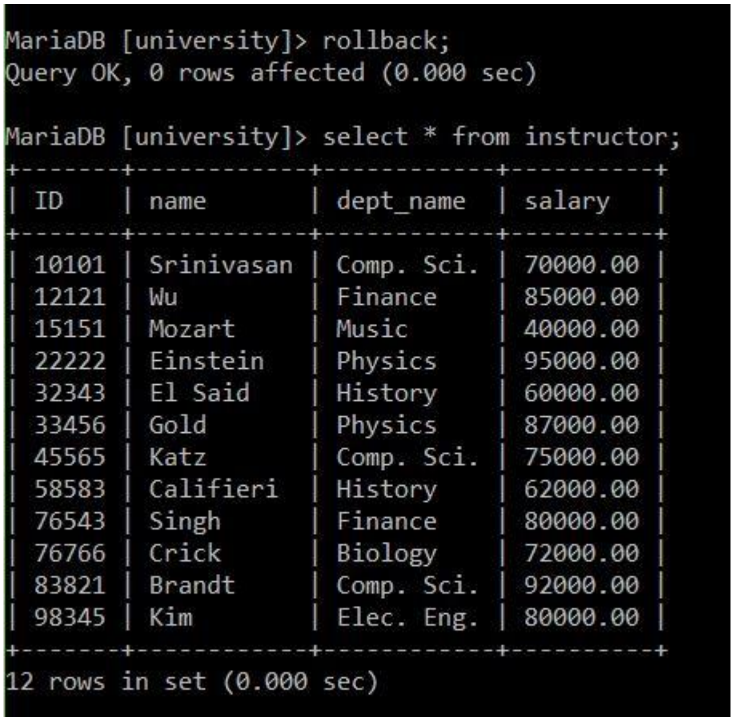
Step 5:- Now let’s see what happens if we commit first and then do rollback.



```
MariaDB [university]> update instructor set salary=salary+5000 where ID=10101;
Query OK, 1 row affected (0.053 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [university]> update instructor set salary=salary-5000 where ID=12121;
Query OK, 1 row affected (0.039 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [university]> commit;
Query OK, 0 rows affected (0.000 sec)

MariaDB [university]> select * from instructor;
+-------+------------+------------+----------+
| ID    | name       | dept_name  | salary   |
+-------+------------+------------+----------+
| 10101 | Srinivasan | Comp. Sci. | 70000.00 |
| 12121 | Wu         | Finance    | 85000.00 |
| 15151 | Mozart     | Music      | 40000.00 |
| 22222 | Einstein   | Physics    | 95000.00 |
| 32343 | El Said    | History    | 60000.00 |
| 33456 | Gold       | Physics    | 87000.00 |
| 45565 | Katz       | Comp. Sci. | 75000.00 |
| 58583 | Califieri  | History    | 62000.00 |
| 76543 | Singh      | Finance    | 80000.00 |
| 76766 | Crick      | Biology    | 72000.00 |
| 83821 | Brandt     | Comp. Sci. | 92000.00 |
| 98345 | Kim        | Elec. Eng. | 80000.00 |
+-------+------------+------------+----------+
12 rows in set (0.000 sec)
```

We can see salaries are changed accordingly and we also commit the operations.

Step 6:- Now, let’s see what happens if we do rollback now.



```
MariaDB [university]> rollback;
Query OK, 0 rows affected (0.000 sec)

MariaDB [university]> select * from instructor;
+-------+------------+------------+----------+
| ID    | name       | dept_name  | salary   |
+-------+------------+------------+----------+
| 10101 | Srinivasan | Comp. Sci. | 70000.00 |
| 12121 | Wu         | Finance    | 85000.00 |
| 15151 | Mozart     | Music      | 40000.00 |
| 22222 | Einstein   | Physics    | 95000.00 |
| 32343 | El Said    | History    | 60000.00 |
| 33456 | Gold       | Physics    | 87000.00 |
| 45565 | Katz       | Comp. Sci. | 75000.00 |
| 58583 | Califieri  | History    | 62000.00 |
| 76543 | Singh      | Finance    | 80000.00 |
| 76766 | Crick      | Biology    | 72000.00 |
| 83821 | Brandt     | Comp. Sci. | 92000.00 |
| 98345 | Kim        | Elec. Eng. | 80000.00 |
+-------+------------+------------+----------+
12 rows in set (0.000 sec)
```

We can see the state of table remains the same the salaries won’t changed to the initial salaries as we committed the update operations, so the latest checkpoint for the table is after the update. Hence, Rollback sent us to the state after we update the values.