Name: Suraj Chandramauli **Roll No: 40**

Exp. No:1 **Date:** 08/06/2021

Familiarization of DDL Commands

Aim: Demonstration of DDL commands:

a) Create a database for the college.

b) Create 2 tables

student and staff tables following fields respectively. with

STUDENT	
Attribute name	**Data type(size)**
Rollno	int(3)
Name	varchar(20)
Age	int(5)
Branch	varchar(10)
semester	int(10)
STAFF	
Attribute name	**Data type(size)**
Eid	int(5)
name	varchar(15)
age	int(5)
branch	varchar(10)
designation	varchar(20)

c) List out the tables present in the college database.

d) Show the structure of student table, staff table.

e) Insert values into student table and staff table (at least 3 rows).

f) Alter the student table by adding a column called 'contact number'(int fileld) and insert values into the added filed.

✔ by droping a coloumn named 'contact number'

✔ modify the existing column named 'semester'

by modidying its data type from 'int' to 'varchar'

by modifying the width of the column from 10 to 5

modifying the constraint of 'semester' colomn from NULL to NOT NULL.

g) Retrieve all data present in student table.

h) Rename student table as 'student details' and staff table as 'staff details'.

i) Delete all data present in the student table and staff table.

j) Drop student table as well as staff table.

k) Drop college database.

Theory:

Data Definition Language (DDL) is a subset of SQL and a part of DBMS(Database Management System). DDL consist of Commands to commands like CREATE, ALTER, TRUNCATE and

DROP. These commands are used to create or modify the tables in SQL

CREATE:

· CREATE is a data definition language (DDL) command that is used for creating database objects such as database and database table.

· It is used for creating database objects like a database and a database table · Syntax for creating a database is as follows :

CREATE DATABASE database_name;

Syntax for creating a table in SQL is as follows:

CREATE TABLE public.customers (column_name_1 datatype [NULL | NOT NULL], column_name_2 datatype [NULL | NOT NULL], . . . column_name_n datatype [NULL | NOT NULL]);

· Example to illustrate database creation in SQL.:

CREATE database practice_db;

Example to illustrate database table creation using the CREATE command: CREATE TABLE public.customer_details (customer_id character varying NOT NULL, customer_name character varying(255) NOT NULL, location character varying(255) NOT NULL, amount_spent numeric NOT NULL, order_id character varying NOT NULL);

ALTER:

· ALTER command in SQL is used to add, rename or modify, drop/delete columns in an existing database table. It can further be used to add and remove various constraints on an existing database table.

· It is used for modifying and renaming elements of an existing database table. · Syntax used for altering a table in SQL by adding a new column is as follows: ALTER TABLE table_name ADD (Columnname_1 datatype);

Syntax used for renaming a table is as follows:

ALTER TABLE table_name_1 RENAME TO table_new_name;

Syntax used for altering a table in SQL by deleting existing columns is as follows: ALTER TABLE table_name DROP columnname_1 , columnname_2, ...;

· Example to add a new column to an existing table:

ALTER TABLE customer_details ADD email_address character varying(255); Example to rename an existing database table:

ALTER TABLE customer_details RENAME TO customer_may;

Example to remove an existing column from a database table:

ALTER TABLE customer_may DROP order_id;

TRUNCATE:

· TRUNCATE TABLE command is used to remove all the data records from the database table. It deletes all the rows permanently. Ergo, we cannot perform a rollback operation to undo a TRUNCATE command.

· It is used to remove all the records from a database table

· Syntax used for writing TRUNCATE command is as follows:

TRUNCATE TABLE table_name;

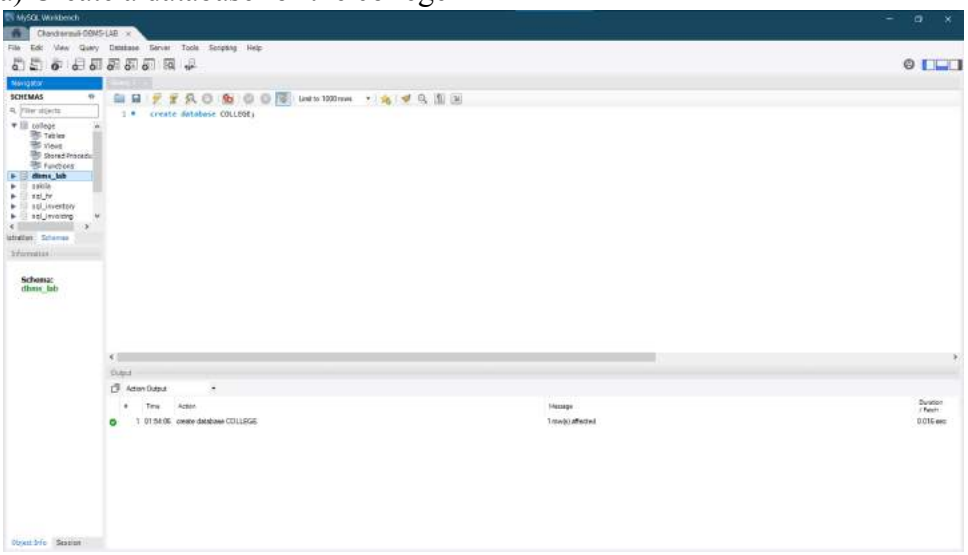
. Example illustrating TRUNCATE command:

Result: As a result DDL commands are familiarized and outputs are verified.

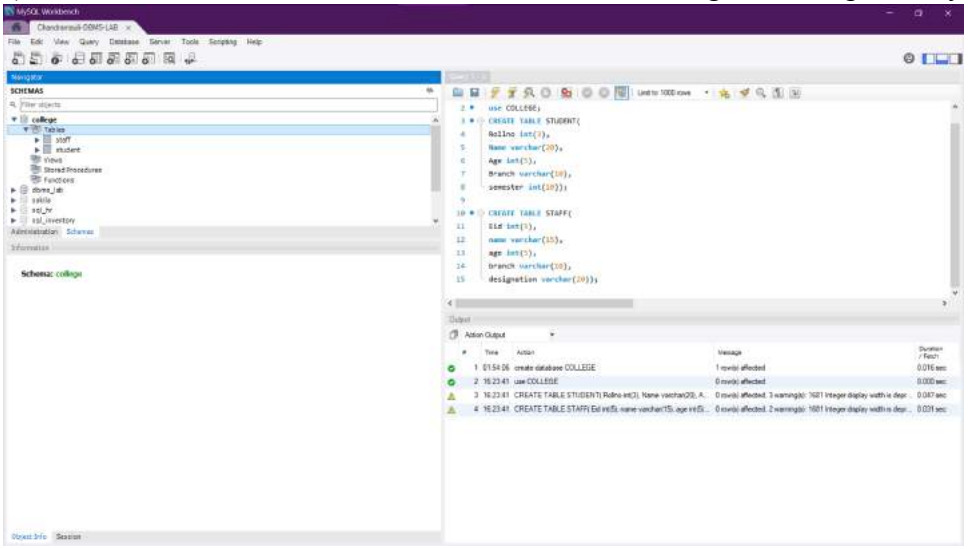
Remarks:(To be filled by faculty)

OUTPUT:

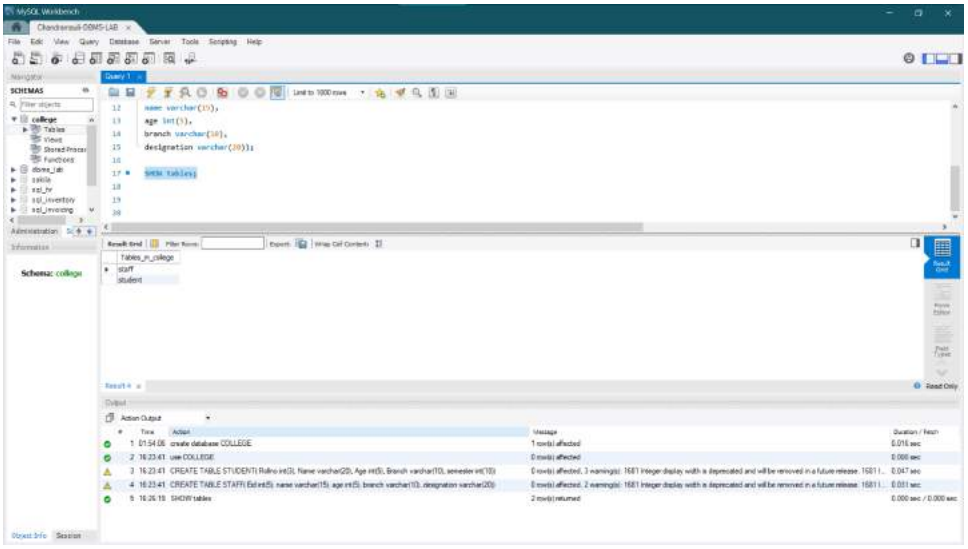
a) Create a database for the college



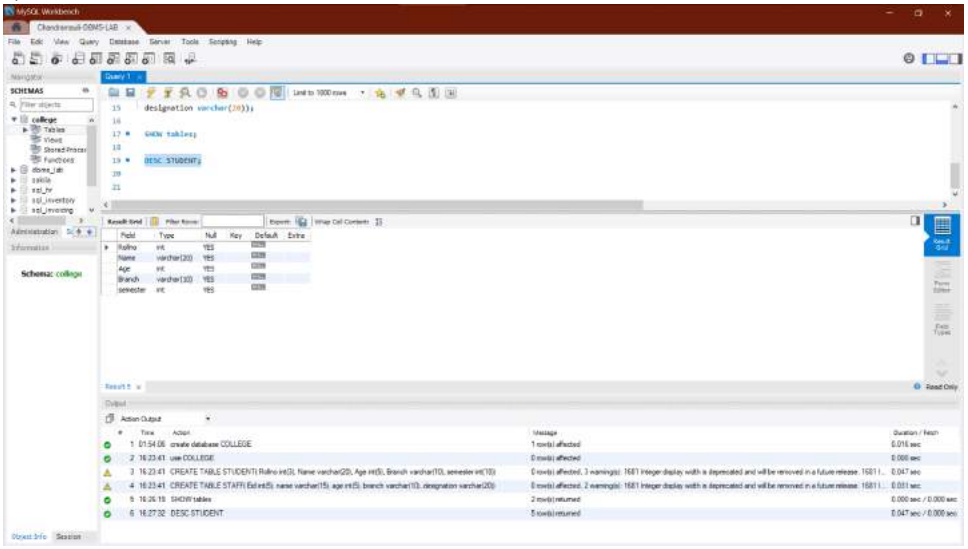
b) Create 2 tables student and staff tables with following fields respectively.

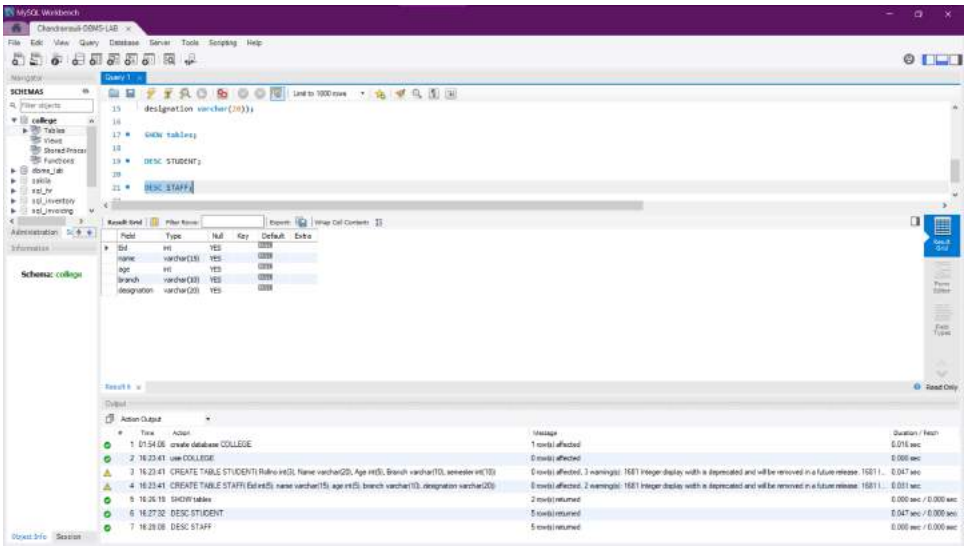


c) List out the tables present in the college database.

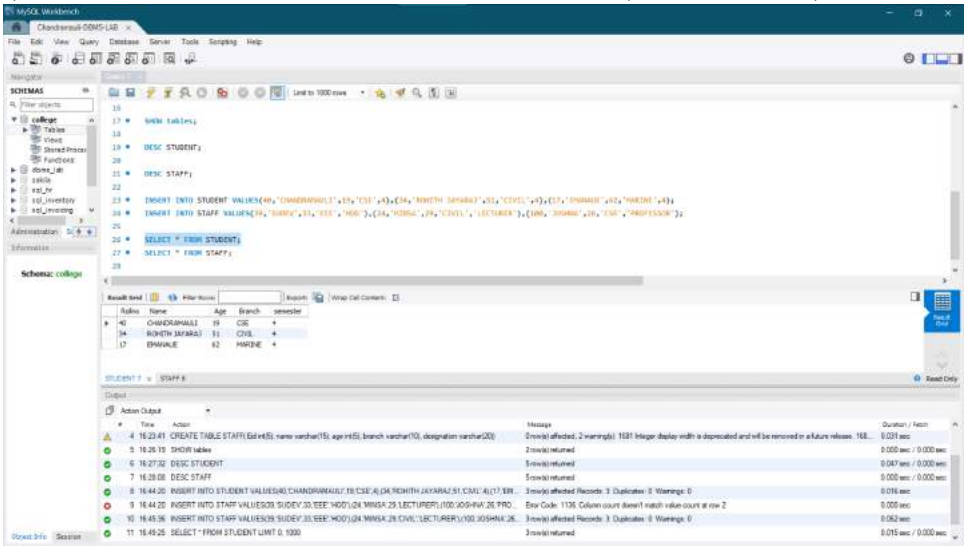


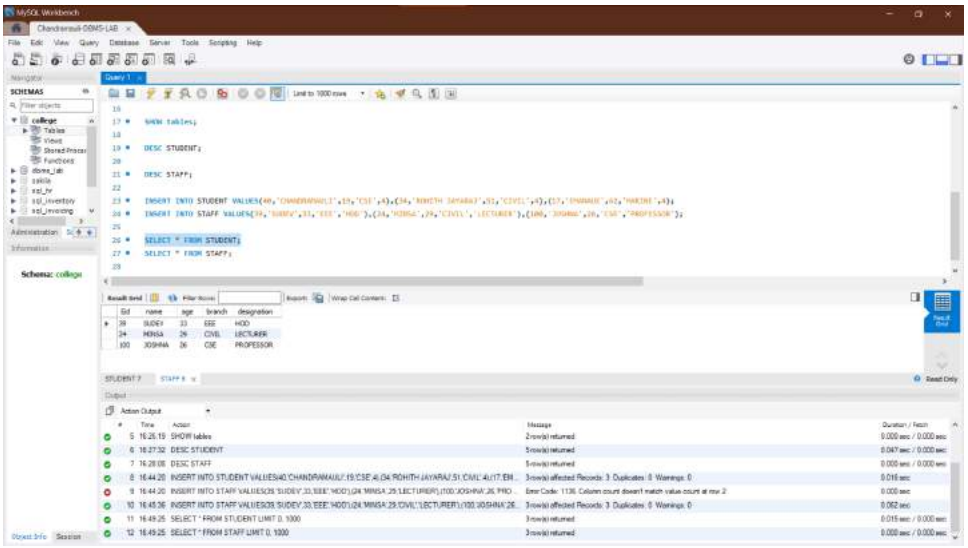
d) Show the structure of student table, staff table.



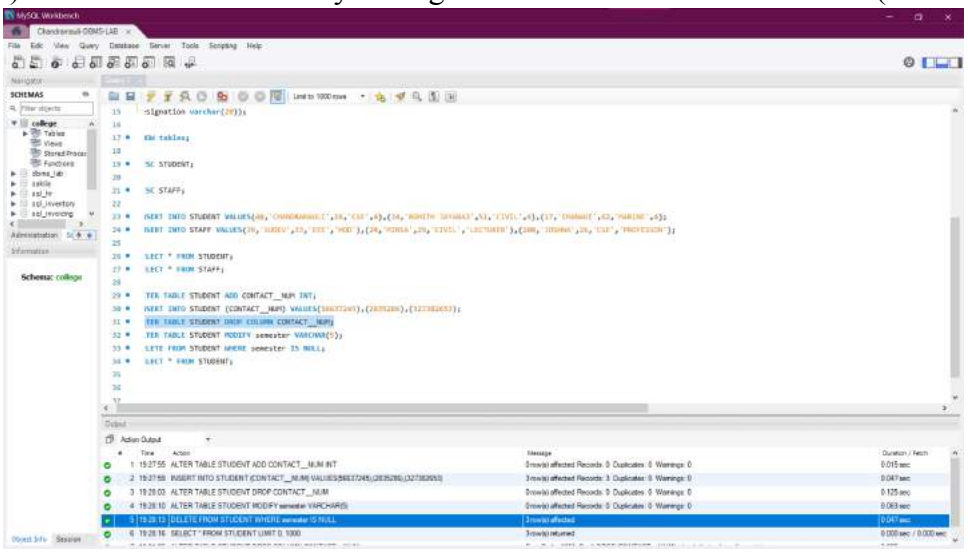


e) Insert values into student table and staff table (at least 3 rows)

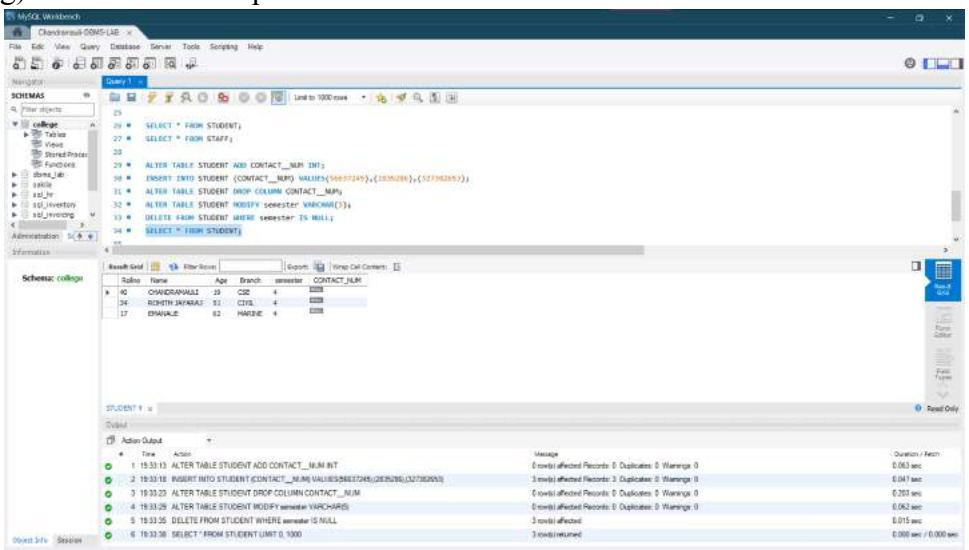




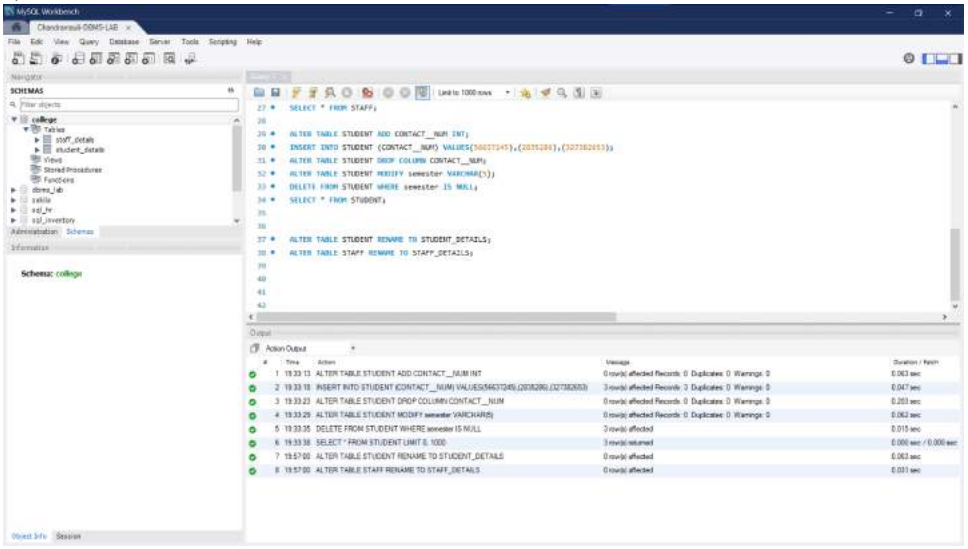
f) Alter the student table by adding a column called 'contact number'(int fileld) and insevalues



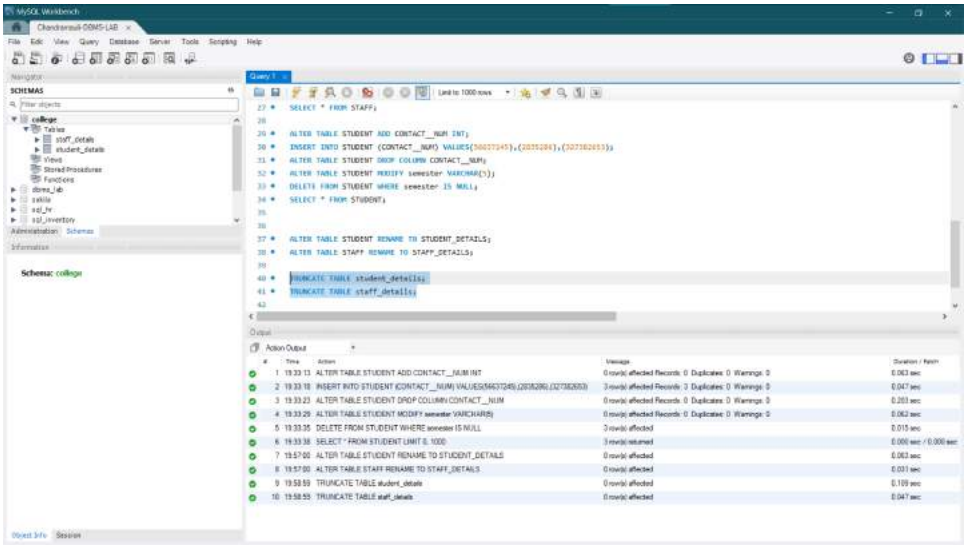
g) Retrieve all data present in student table



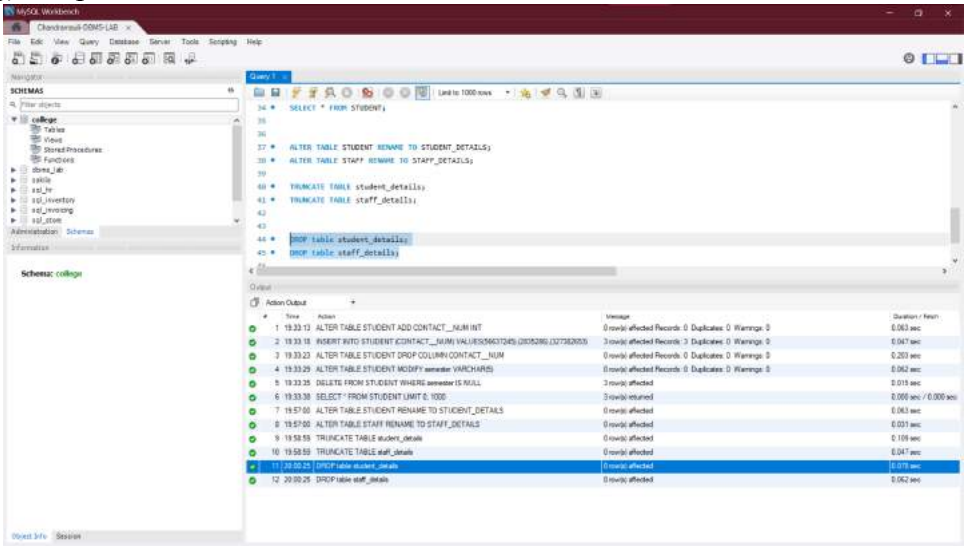
h) Rename student table as 'student details' and staff table as 'staff details'



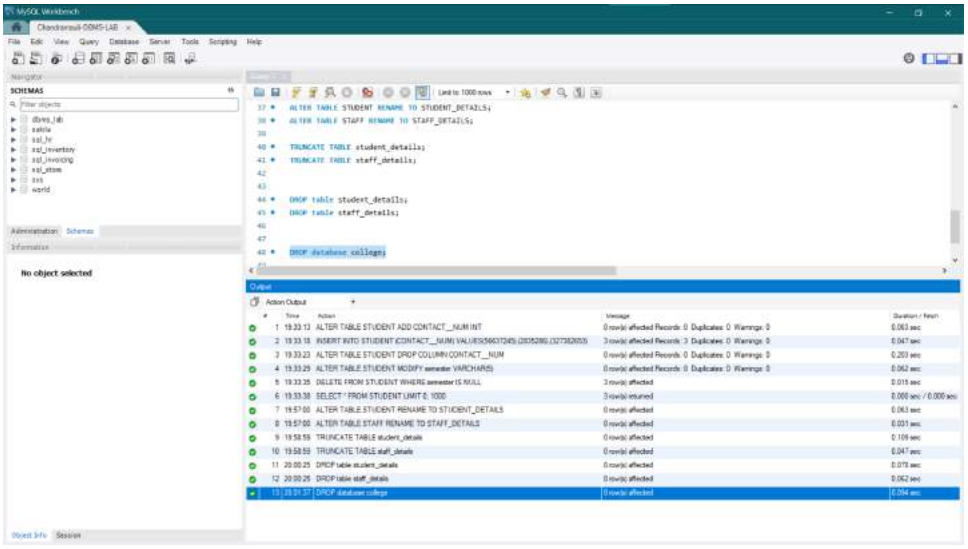
i) Delete all data present in the student table and staff table



j) Drop student table as well as staff table.



k) Drop college database



(Left side of a page)

Name: Suraj Chandramauli Rollno:40

Exp no 2 **Date : 08/06/2021**

Familiarization of DML Commands

Aim: Demonstration of DML commands:
Create 2 tables employee and department with the corresponding field and constraints given below.

EMPLOYEE	
Eno	primary key and first letter is 'E'
Ename	NOT NULL
Salary	should not be zero
DNO	foreign key referencing DNO of DEPARTMENT
DOJ	
MNGRNO	
JOB	
ADDRESS	
CITY	values must be 'cochin', 'Bombay', 'madrass', 'Delhi'
PINCODE	

where Eno=employee number, dno=department number, mngrno=manager number, doj=date of joing

DEPARTMENT	
DNO	PRIMARY KEY
DNAME	NOT NULL
CNT_EMP	should not be greater than 15
DEPT_HOD	

Here CNT_EMP= employee count, DEPT_HOD= head of the department

a) insert values into employee and department tables

EMPLOYEE									
Eno	**Ename**	**Salary**	**Dno**	**DOJ**	**MNGRNO**	**Job**	**Address**	**city**	**PIN**
E10	Lini	40000	D10	1/1/1990	12	Sales	Vytila	Cochin	048622
E20	Anu	50000	D20	30/9/1998	33	Commerce	Kollam	Bombai	665356
E15	Giri	60000	D30	1/9/1999	12	Sales	Kerala	Delhi	62235

E16	Lulu	50000	D15	1/9/1997	-	Agriculture	Kerala	Madras	64633
E12	Sini	40000	D10	1/1/1998	22	finance	Alappuzha	Delhi	241156

DEPARTMENT			
Dno	**Dname**	**CNT-EMP**	**Dept-HOD**
D10	Sales	2	Sreela
D20	Agriculture	1	Vinod
D15	Finance	1	Sreeni
D30	Commerce	1	Greena

b) Display all the employee details & department details.

c) update the 'city' and 'salary' of emplyee whose Eid=E12 to 'cochin' and '70000'. d) Display all the employee details & department details.
e) List the name of employees joined after 1-1-1998 and working in department number d10. f) List all employees working in department other than department number d30. g) List the name of employees working in department 'sales'.
h) List the name of employee who does not have a manager.

i) Display employees details whose city=’cochin’.

j) List the HOD’s of different department.

k) Find out who is the HOD of department D20.

l) Delete employee whose Eid=E15 from employee table.

m) Display details of employee table.

n) Delete employees whose city=’Delhi’.

o) Display details of employee table.

p) Delete all the employees from employee table.

q) Display details of employee table.

Theory:

The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements like SELECT, INSERT, UPDATE , DELETE.

SELECT:

SELECT command or statement in SQL is used to fetch data records from the database table and present it in the form of a result set. It is usually considered as a DQL command but it can also be considered as DML.

· It is used to query or fetch selected fields or columns from a database table · Syntax for writing a SELECT query in SQL is as follows :

SELECT column_name1, column_name2, … FROM table_name WHERE condition_ expression;

· Examples to illustrate the use of SELECT command.:

SELECT customer_id, sale_date, order_id, store_state FROM customers;

INSERT:

· INSERT commands in SQL are used to insert data records or rows in a database table. In an INSERT statement, we specify both the column_names for which the entry has to be made along with the data value that has to be inserted.

· It is used to insert new data records or rows in the database table.

· Syntax for writing INSERT statements in SQL is as follows:

INSERT INTO table_name (column_name_1, column_name_2, column_name_3, ...) VALUES (value1, value2, value3, ...);

· Examples to further illustrate the INSERT statement.:

INSERT INTO public.customers(customer_id, sale_date, sale_amount, salesperson, store_state, order_id) VALUES (1005,'2019-12-12',4200,'R K Rakesh','MH','1007');

UPDATE:

· UPDATE command or statement is used to modify the value of an existing column in a database table

· It is used to set the value of a field or column for a particular record to a new value. · Syntax for writing an UPDATE statement is as follows :

UPDATE table_name SET column_name_1 = value1, column_name_2 = value2, ... WHERE condition;

· Example based on the UPDATE statement in SQL:

UPDATE customers SET store_state = 'DL' WHERE store_state = 'NY';

DELETE:

· DELETE statement in SQL is used to remove one or more rows from the database table. It does not delete the data records permanently. We can always perform a rollback operation to undo a DELETE command. With DELETE statements we can use the WHERE clause for filtering specific rows.

· It is used to remove one or more rows from the database table

· Syntax for writing an DELETE statement is as follows :

DELETE FROM table_name WHERE condition;

· Example based on the DELETE command in SQL:

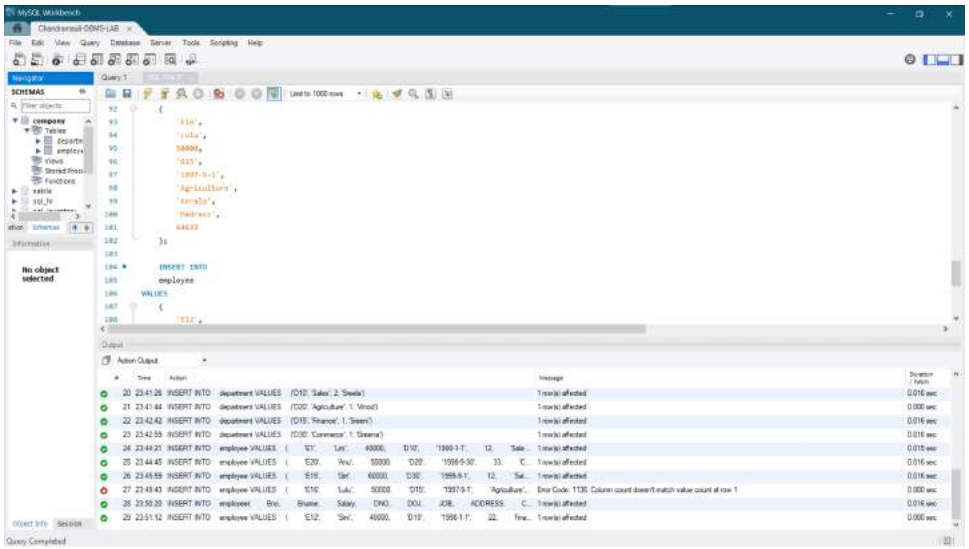
DELETE FROM customers WHERE store_state = 'MH' AND customer_id = '1001';

Result: As a result DML commands are familiarized and outputs are verified.

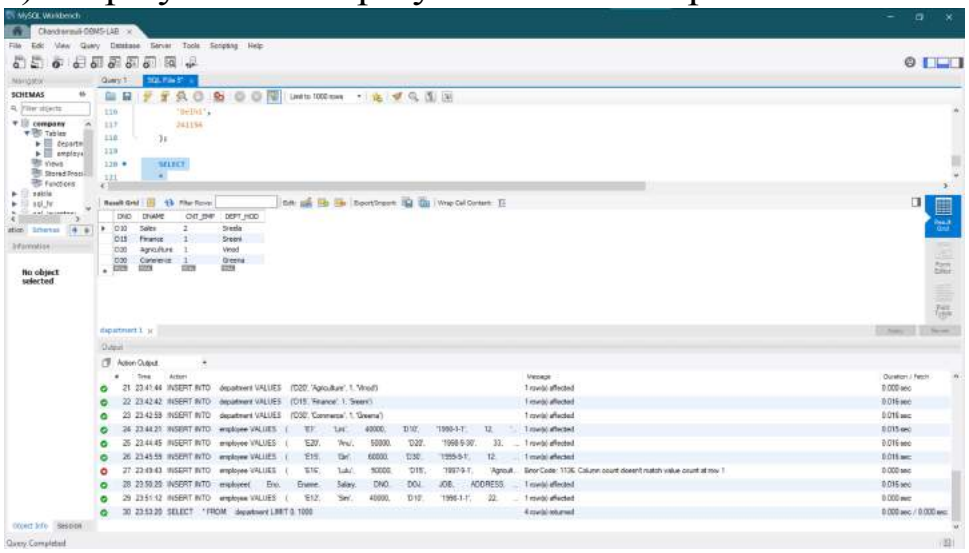
Remarks:(To be filled by faculty)

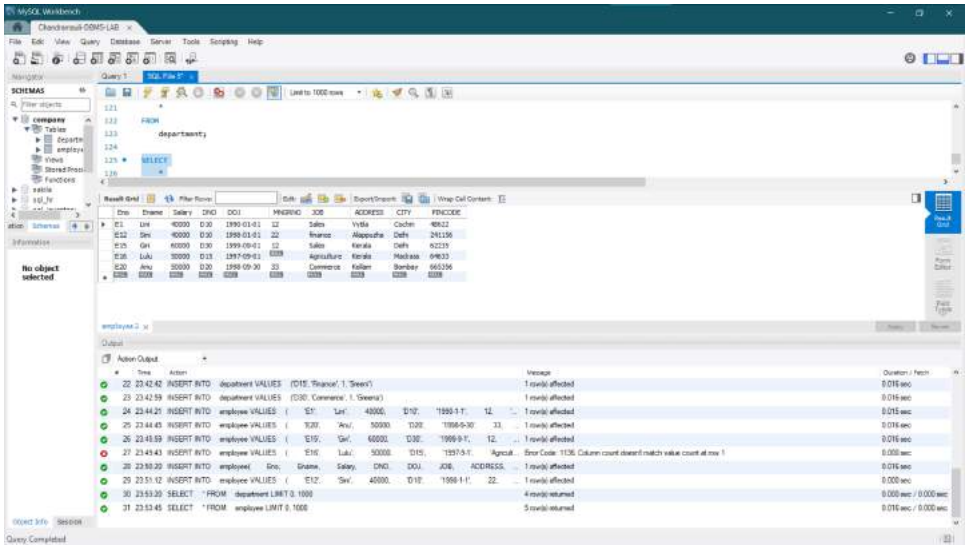
OUTPUT:

a) insert values into employee and department tables

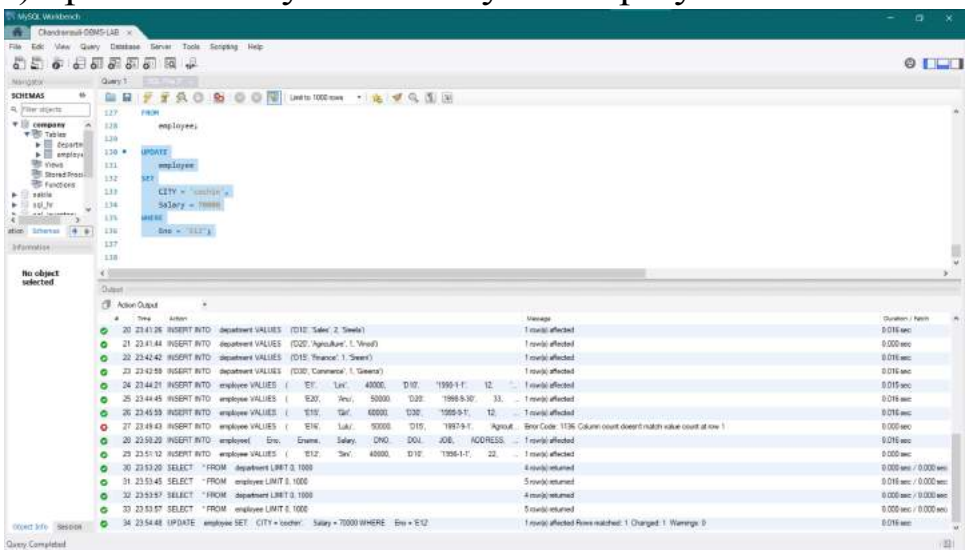


b) Display all the employee details & department details.

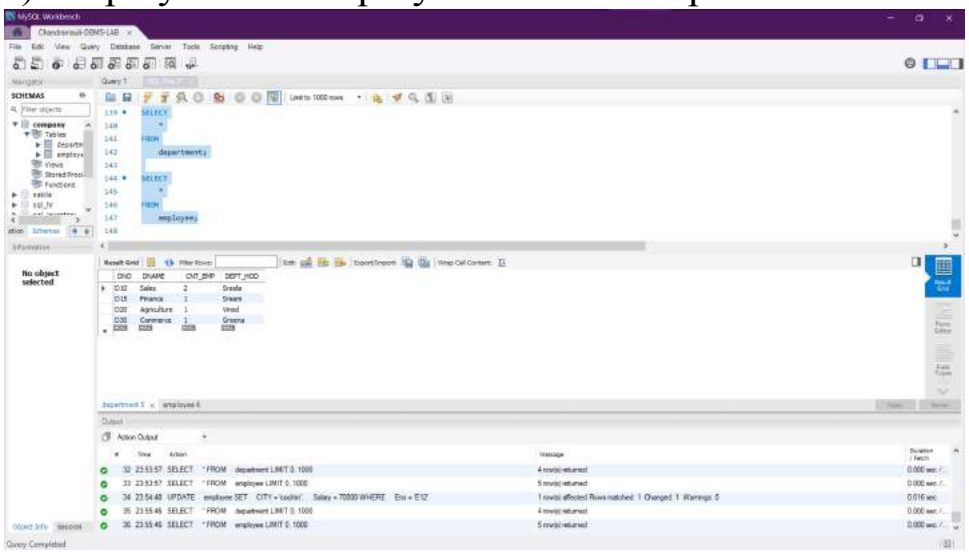


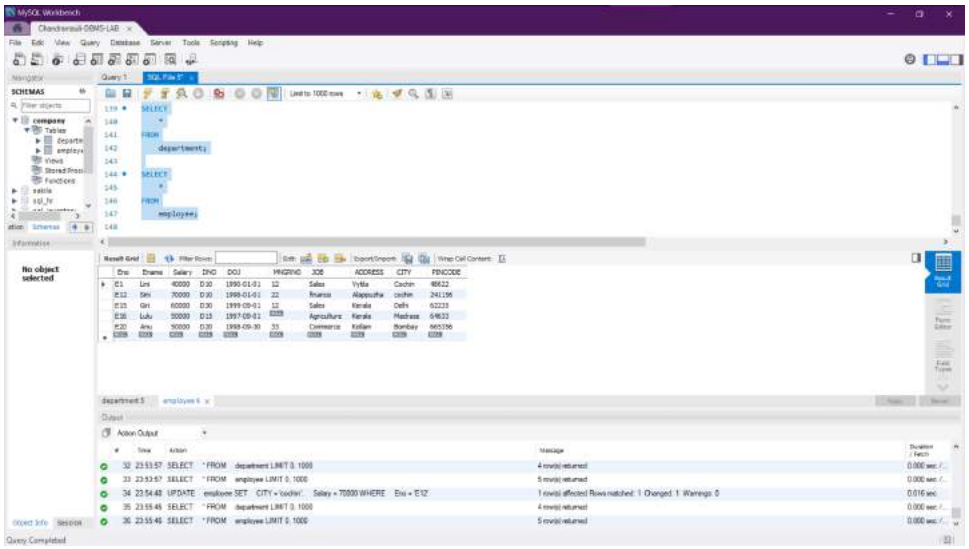


c) update the 'city' and 'salary' of employee whose Eid=E12 to 'cochin' and '70000'.

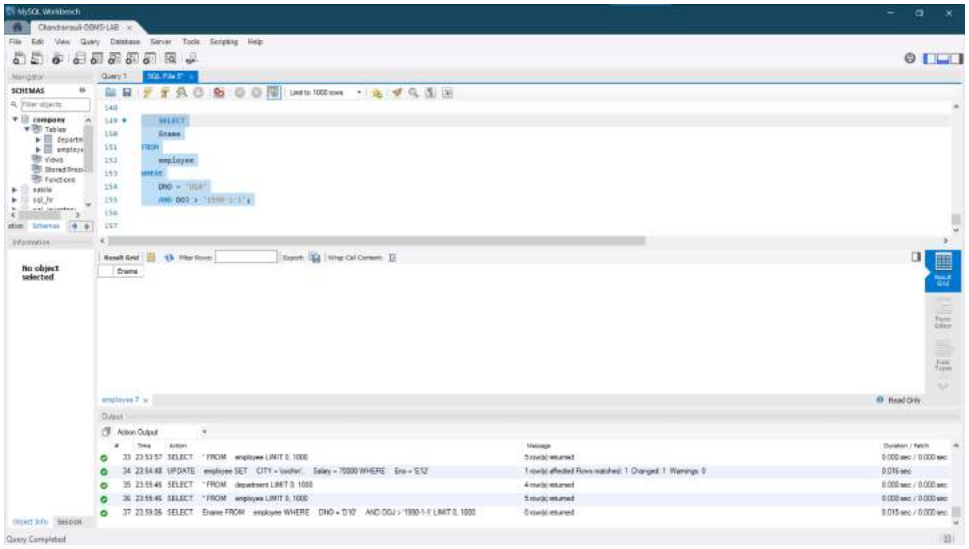


d) Display all the employee details & department details

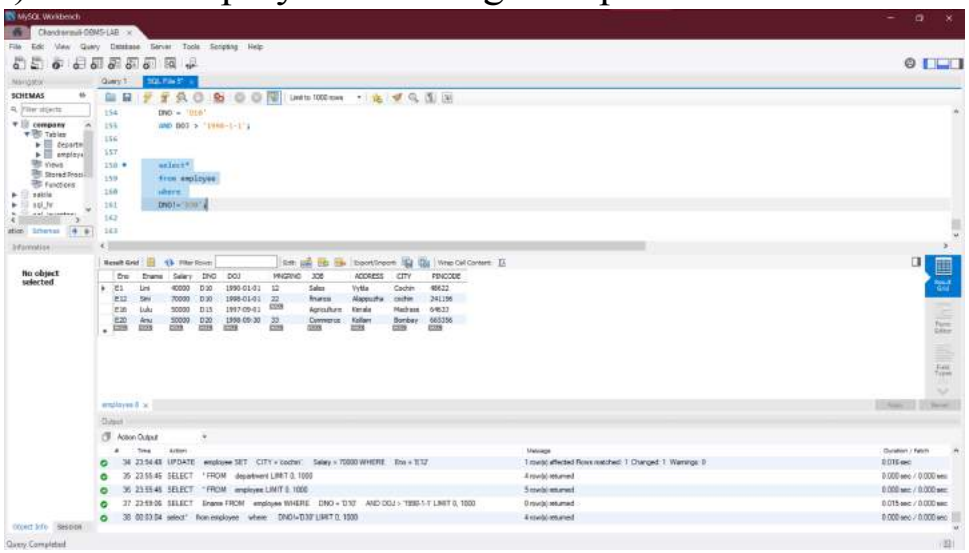




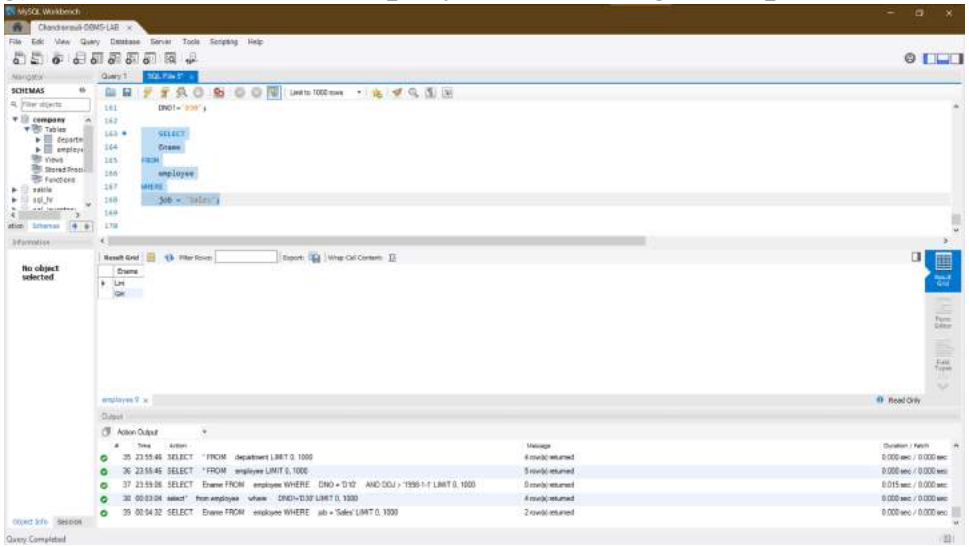
e) List the name of employees joined after 1-1-1998 and working in department number d10.



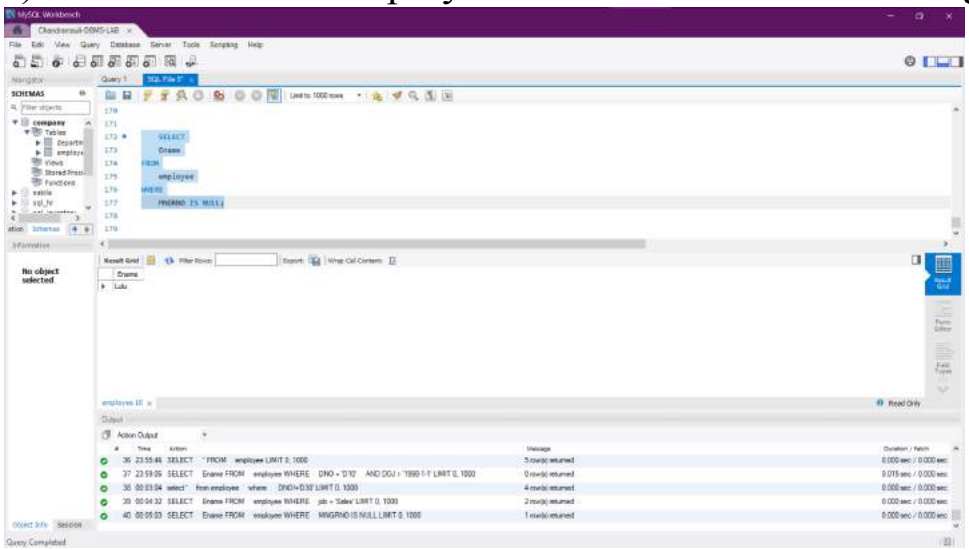
f) List all employees working in department other than department number d30



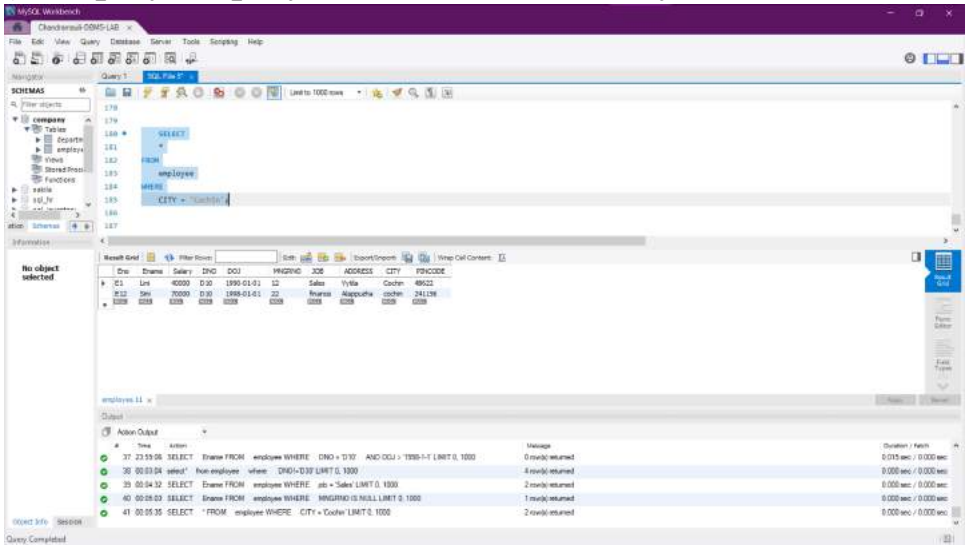
g) List the name of employees working in department 'sales'.



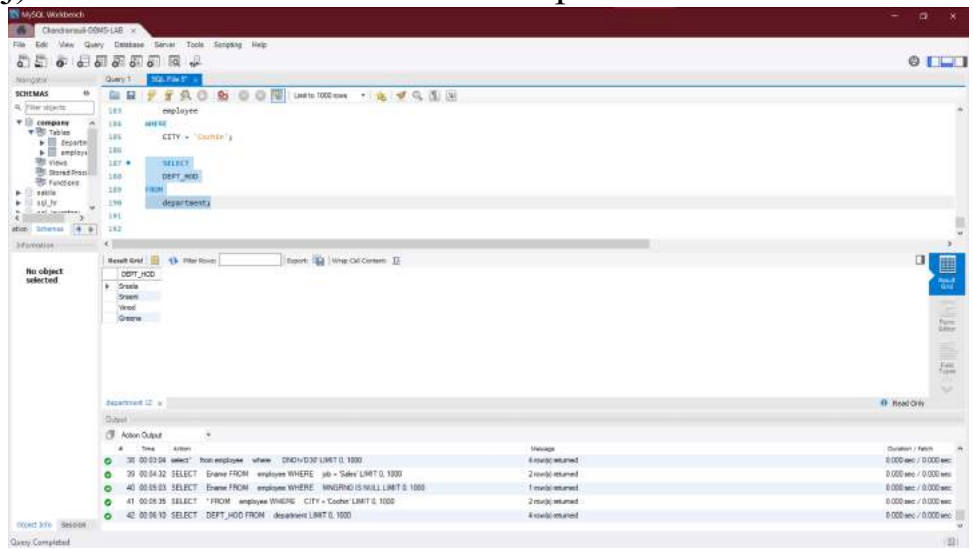
h) List the name of employee who does not have a manager.



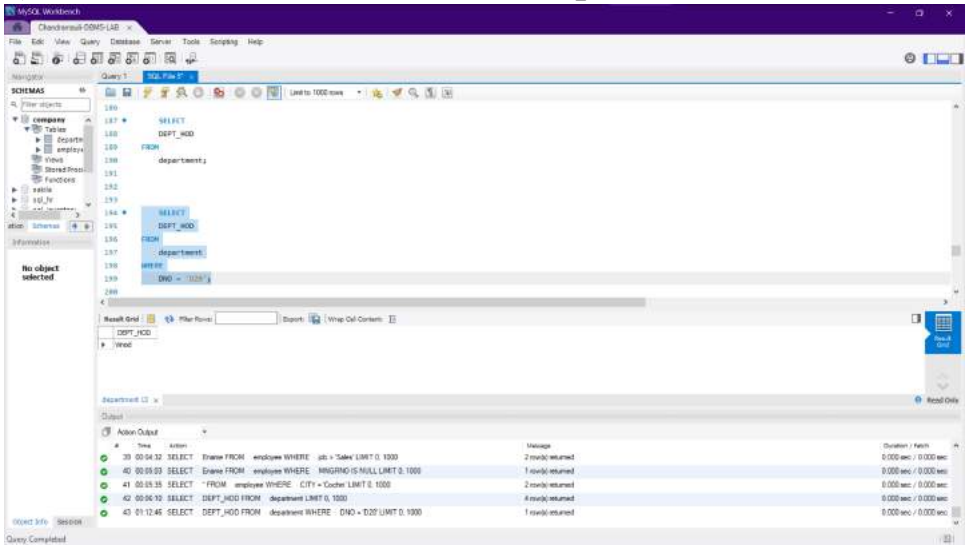
i) Display employees details whose city=’cochin’



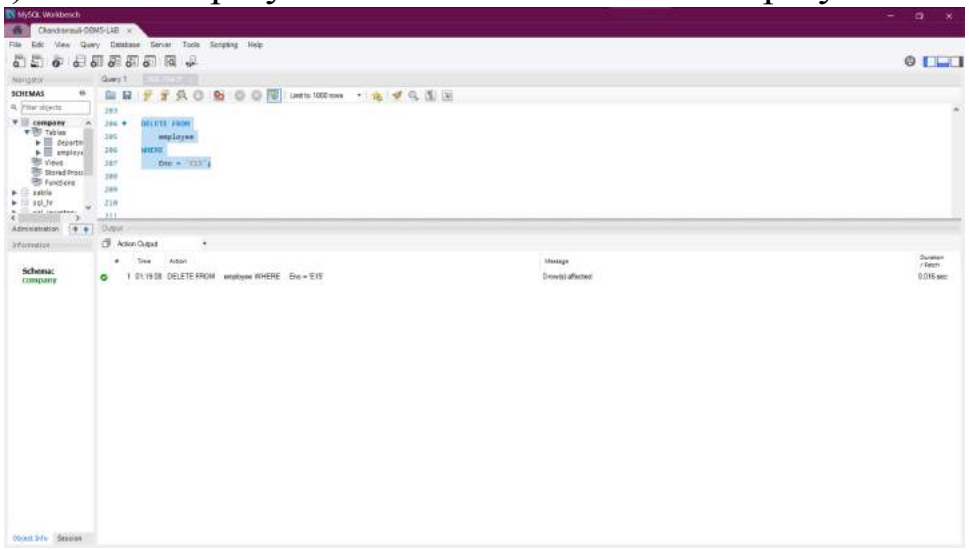
j) List the HOD’s of different department



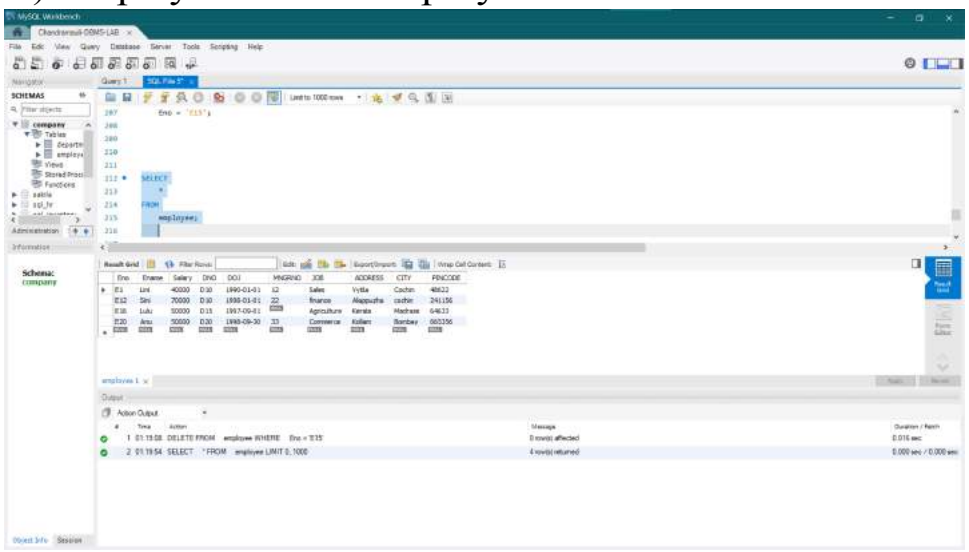
k) Find out who is the HOD of department D20



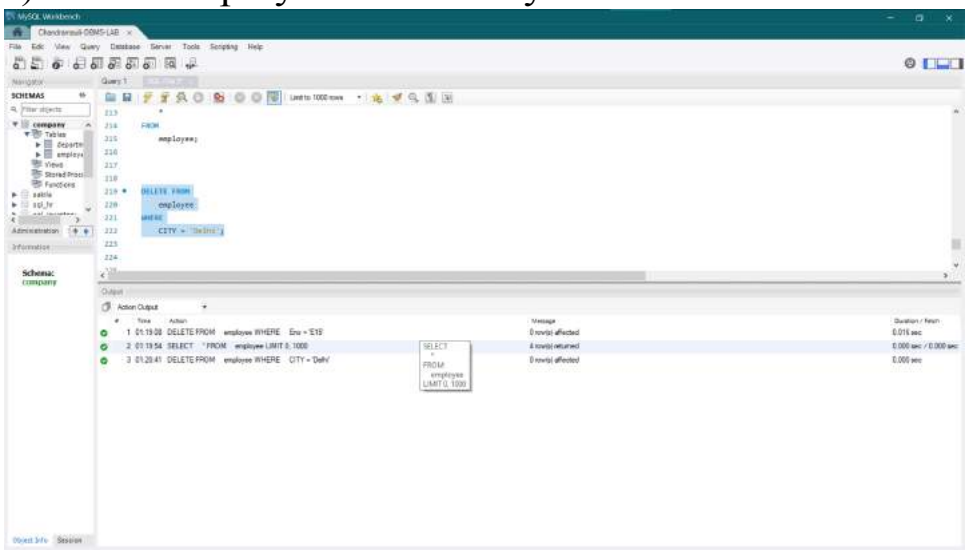
l) Delete employee whose Eid=E15 from employee table.



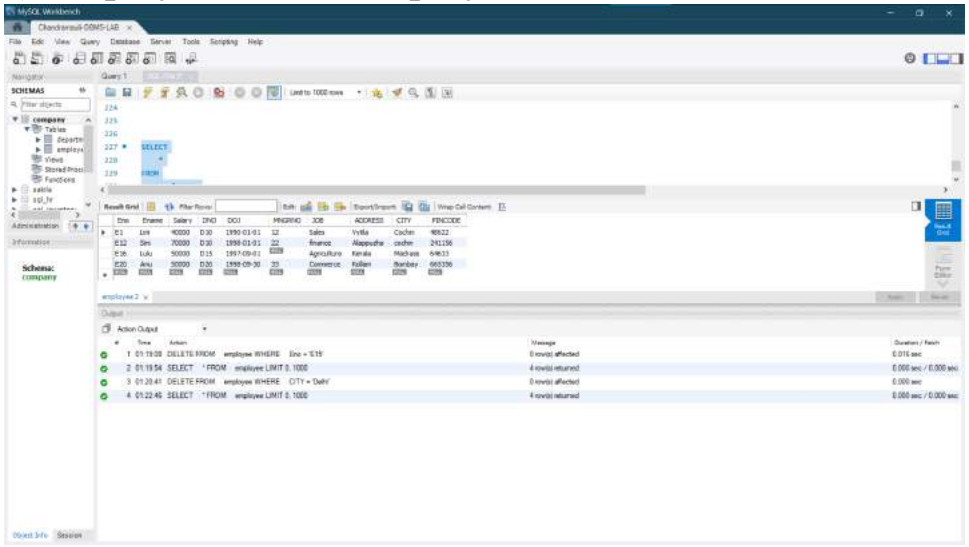
m) Display details of employee table.



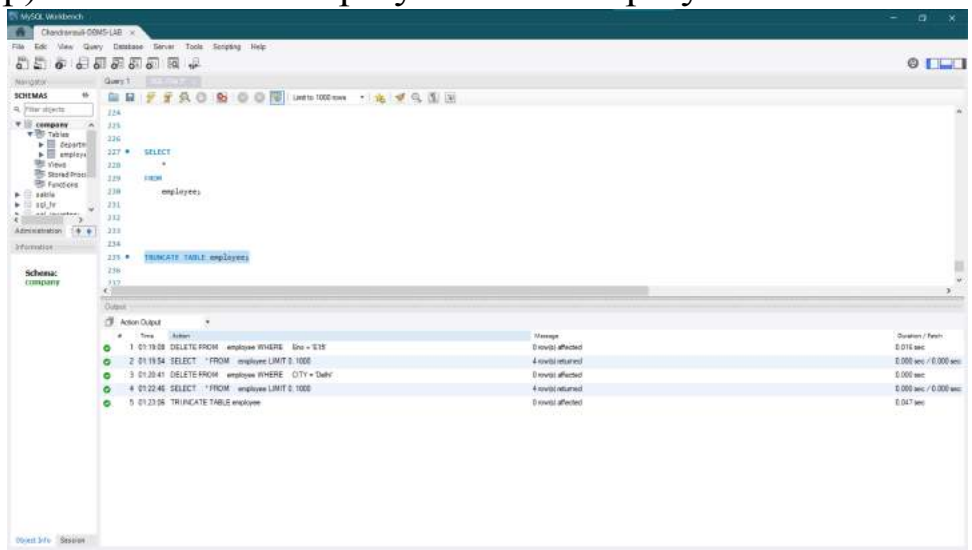
n) Delete employees whose city=’Delhi’.



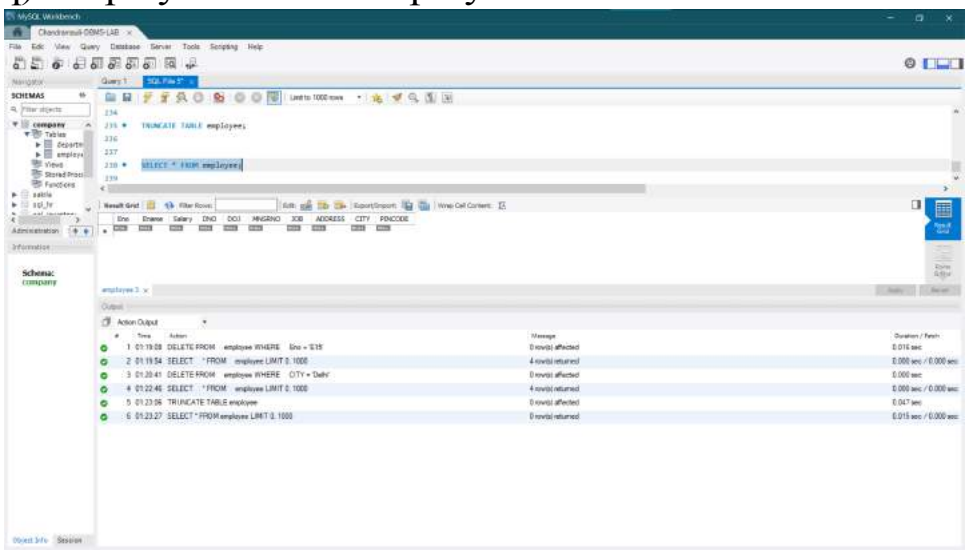
o) Display details of employee table.



p) Delete all the employees from employee table.



q) Display details of employee table.



(Left side of a page)

Name: Suraj Chandramauli **Roll No: 40**

Exp. No: 3 **Date:** 08/06/2021

Familiarization of TCL Commands

Aim: Demonstration of TCL commands:

Create a database for bank & create a table with name ‘savings-account’. The fields are CID, cname, balance, date of joining.

a) Add 2 records to the ‘savings-account’ table.

b) Display the values of ‘savings-account’ table.

c) Make the changes permanently.

d) Add 2 more records to the 'savings-account' table.

e) Display all the records of 'savings-account' table.

f) Modify the balance amount by adding the interest of 6%..

g) Display all the records of 'savings-account' table.

h) Abandon the last changes.

i) Display all the records of 'savings-account' table.

j) Add a marker to the changed state as 'A'.

k) Add two more records to the ''savings-account' table.

l) Display all the records of 'savings-account' table.

m) Modify the balance amount by adding the interest of 6%.

n) Display all the records of 'savings-account' table.

o) Add a marker to the changed state as 'B'.

p) Delete one record from the 'savings-account table.

q) Display all the records of 'savings-account' table.

r) Abandon the last deletion (ie, recover the table with deleted row).

s) Display all the records of 'savings-account' table.

t) Abandon to save point/marker 'A'.

u) Display all the records of 'savings-account' table.

Theory:

Transaction Control Language (TCL) Command are used to manage transaction in database such as COMMIT,ROLLBACK,SAVEPOINT.

COMMIT: - · COMMIT command is used to make a transaction permanent in a database. So it can be said that commit command saves the work done as it ends the current transaction by making permanent changes during the transaction.

· COMMIT command saves all the work done.

· Syntax for COMMIT command in SQL is as follows:

```
COMMIT;
```

· Example for COMMIT command in SQL is as follows:

```
UPDATE EMPLOYEE SET City = 'Cochin';
COMMIT;
```

ROLLBACK:

· ROLLBACK command is used to restore the database to its original state since the last command that was committed.

· ROLLBACK command restores database to original since the last COMMIT. · Syntax for ROLLBACK command in SQL is as follows:

```
ROLLBACK;
```

Syntax for the ROLLBACK command is used along with savepoint command to leap to a save point in a transaction:

ROLLBACK TO <savepoint_name>;

· Example for ROLLBACK command in SQL is as follows:

UPDATE EMPLOYEE SET City= 'Bangalore';

ROLLBACK;

Example for the ROLLBACK command is used along with savepoint command to leap to a save point in a transaction is as follows:

ROLLBACK TO A;

SAVEPOINT:

· SAVEPOINT command is used to save the transaction temporarily. So the users can rollback to the required point of the transaction.

· SAVEPOINT command is used for saving all the current point in the processing of a transaction.

· Syntax for SAVEPOINT command in SQL is as follows:

SAVEPOINT savepoint_name;

· Example for SAVEPOINT command in SQL is as follows:

INSERT INTO STUDENT VALUES (813, 'TRIKKZ');

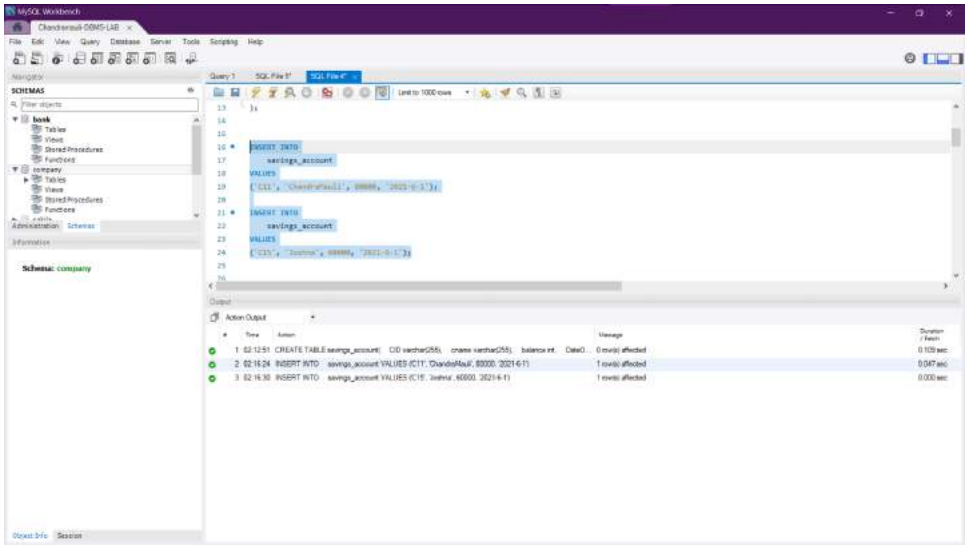
SAVEPOINT B;

Result: As a result TCL commands are familiarized and outputs are verified.

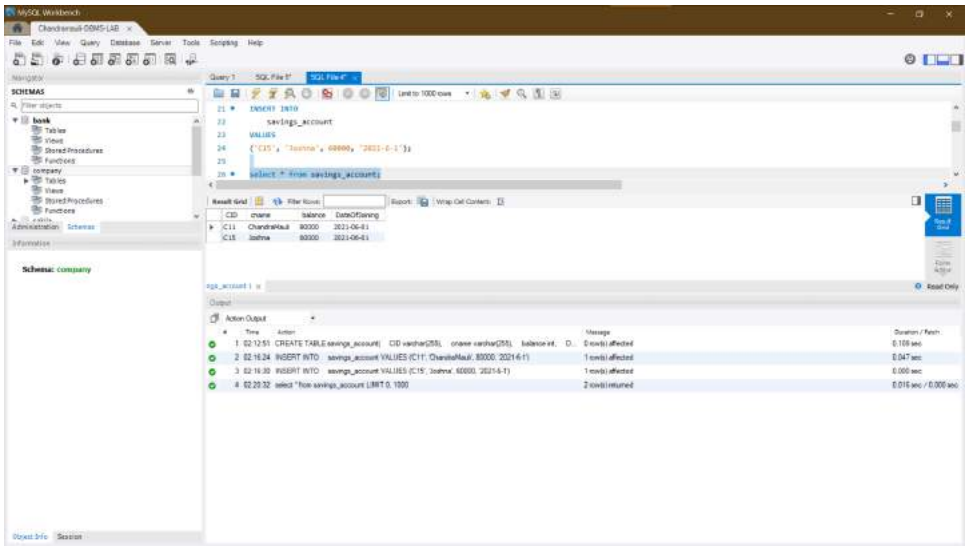
Remarks:(To be filled by faculty)

OUTPUT :

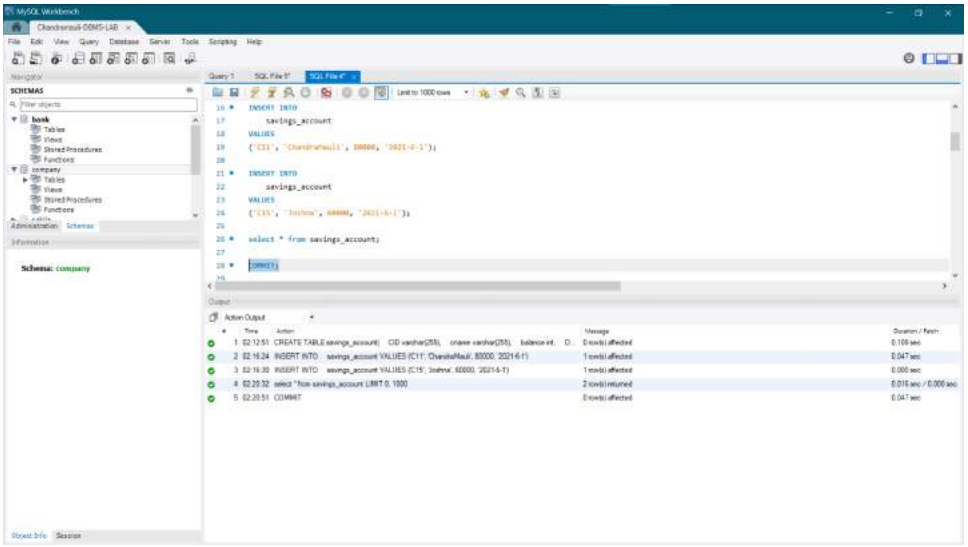
a)Add 2 records to the ‘savings-account’ table



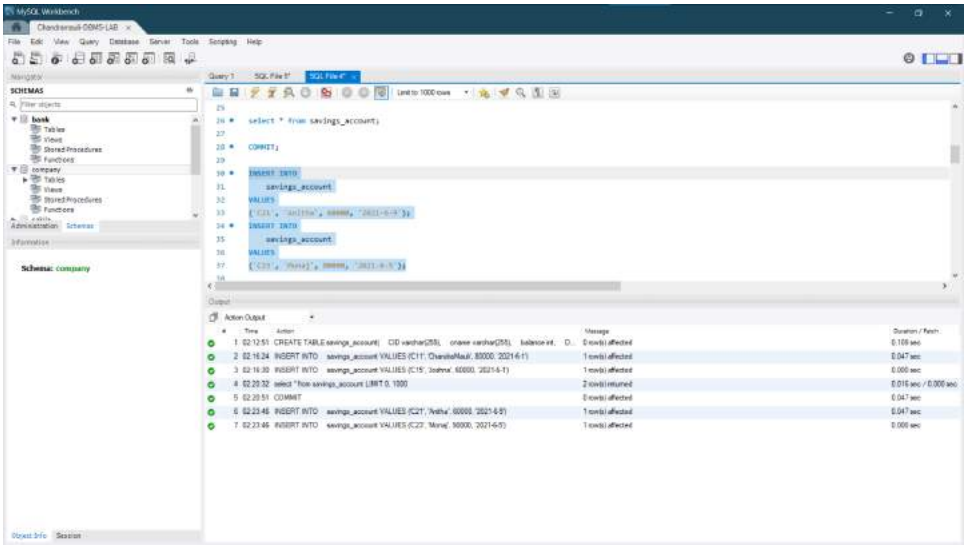
b) Display the values of ‘savings-account’ table



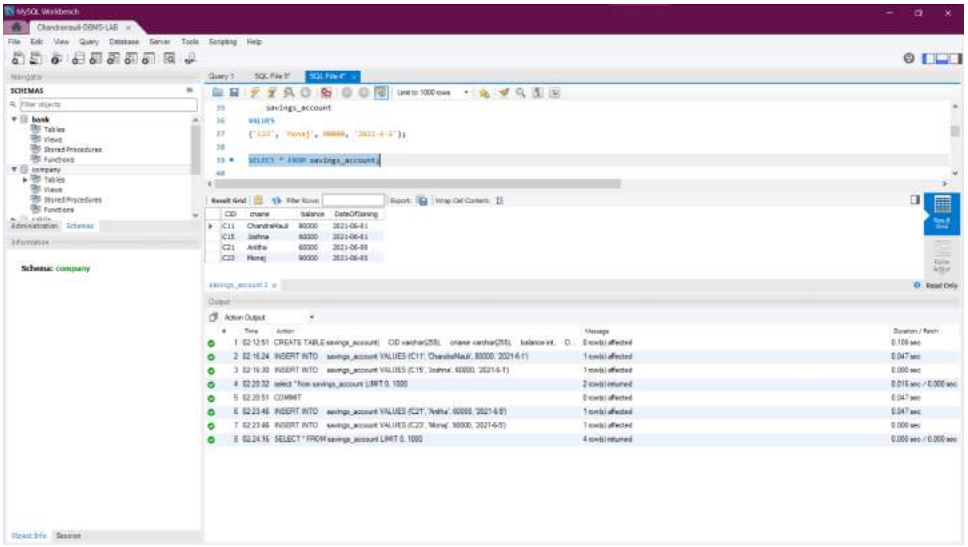
c) Make the changes permanently.



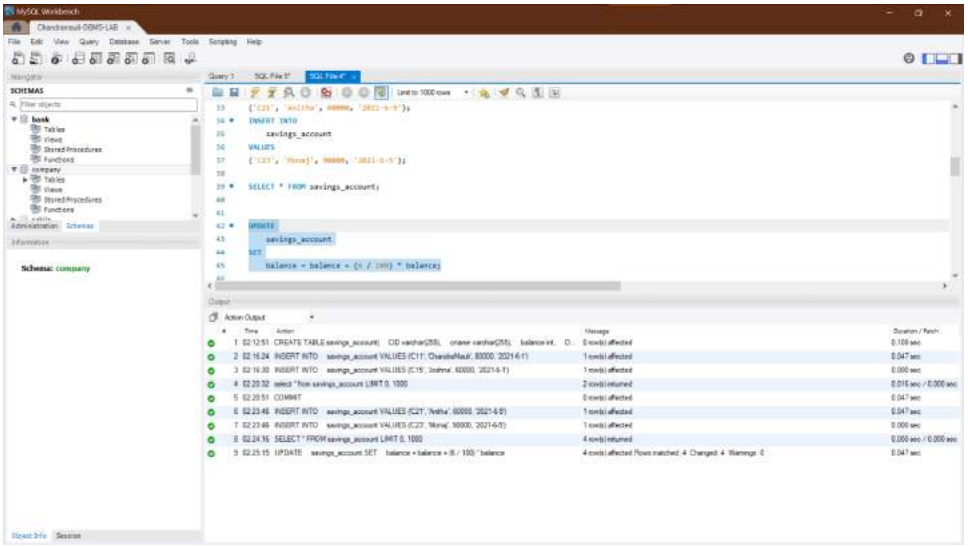
d) Add 2 more records to the ‘savings-account’ table



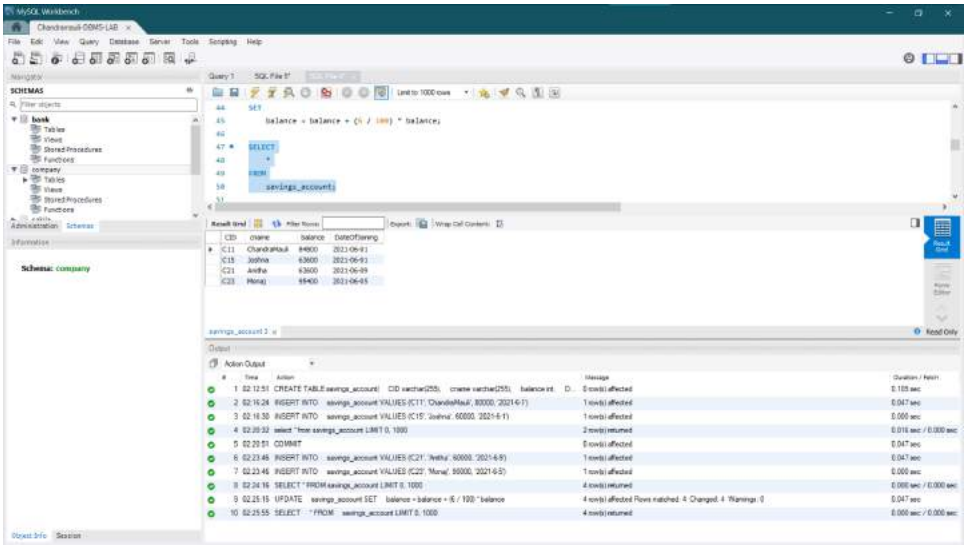
e) Display all the records of ‘savings-account’ table.



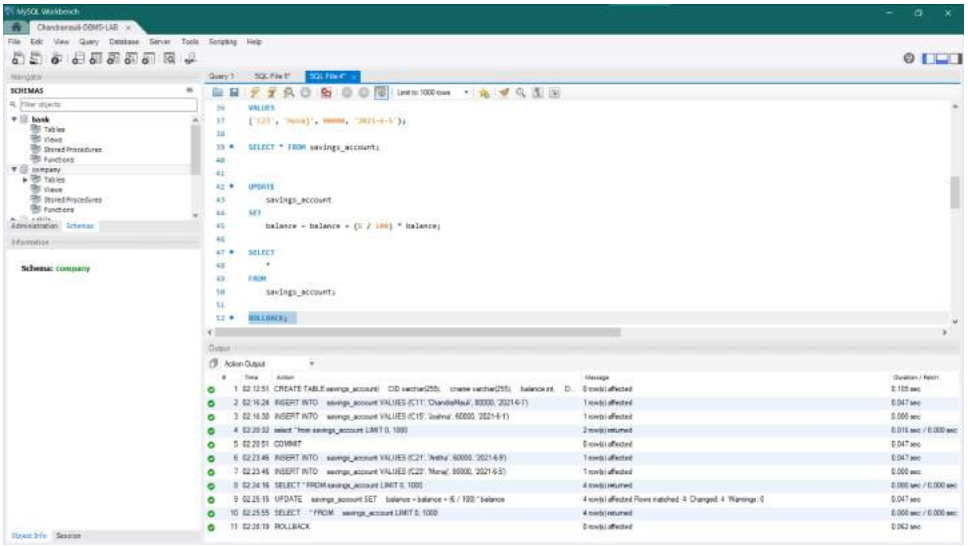
f) Modify the balance amount by adding the interest of 6%



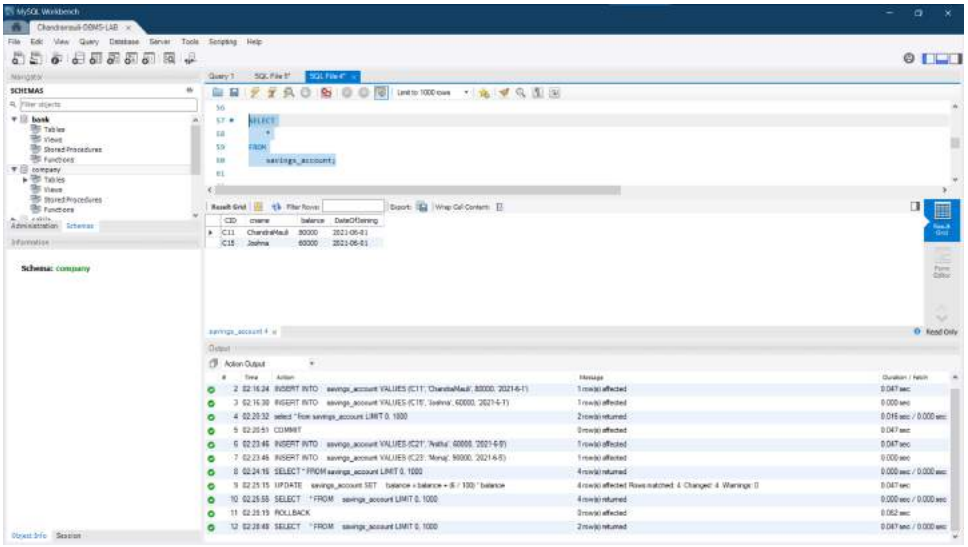
g) Display all the records of ‘savings-account’ table.



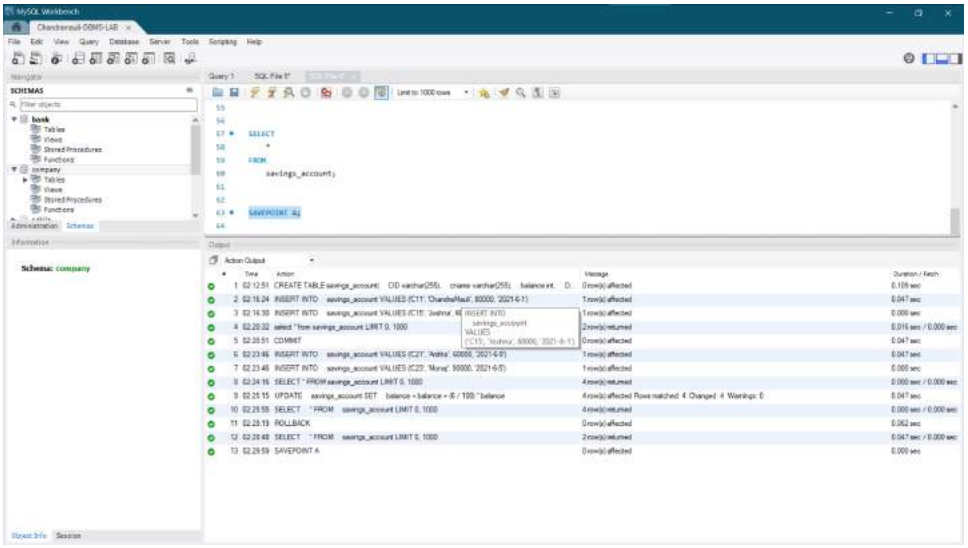
h) Abandon the last changes.



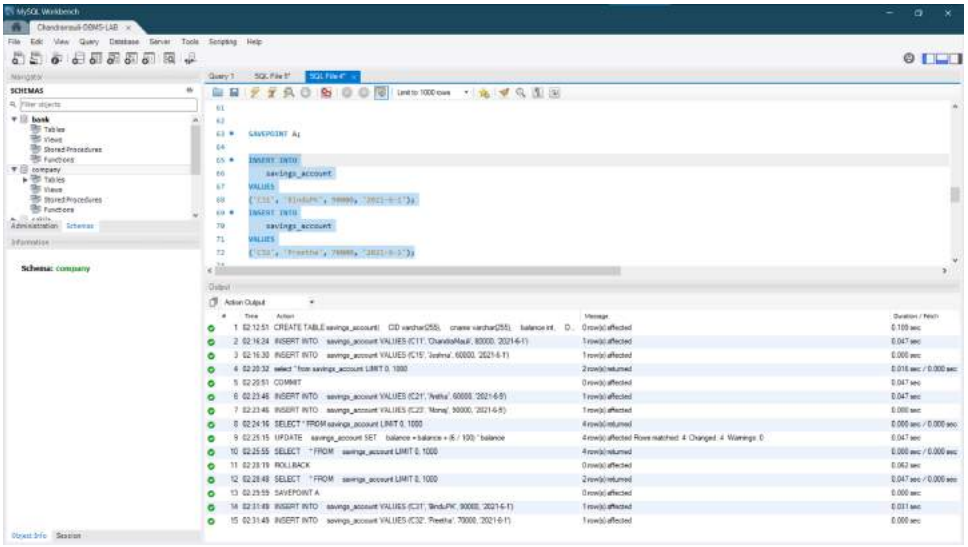
i) Display all the records of ‘savings-account’ table



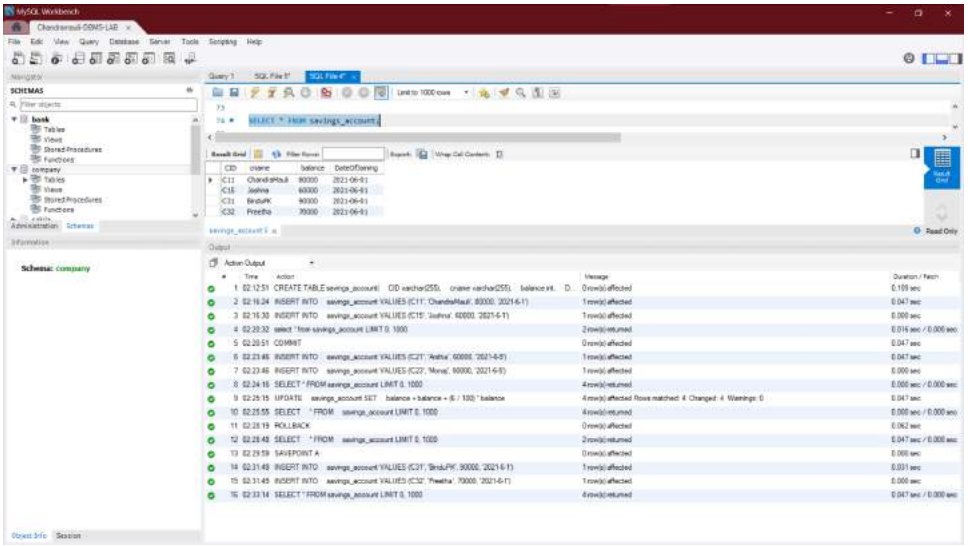
j) Add a marker to the changed state as ‘A’.



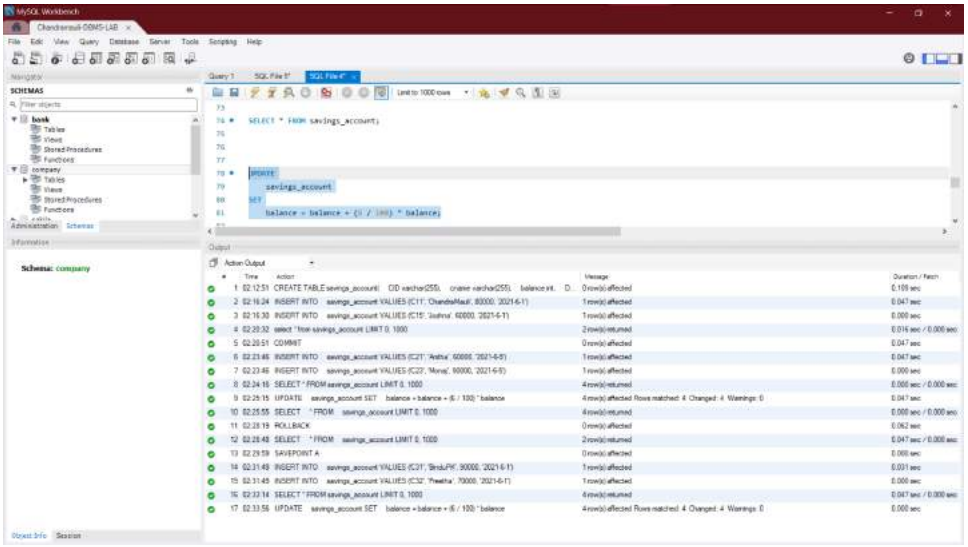
k) Add two more records to the ‘’savings-account’ table.



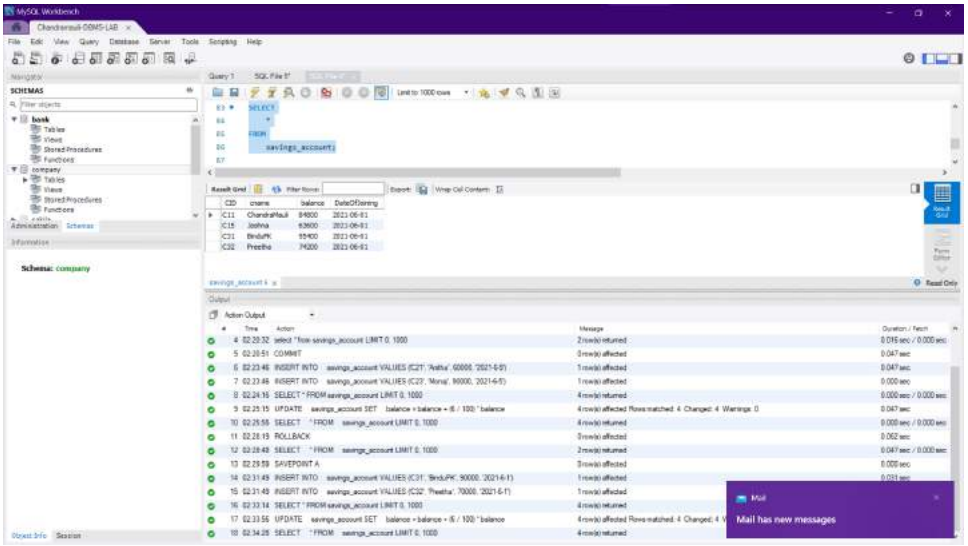
l) Display all the records of ‘savings-account’ table.



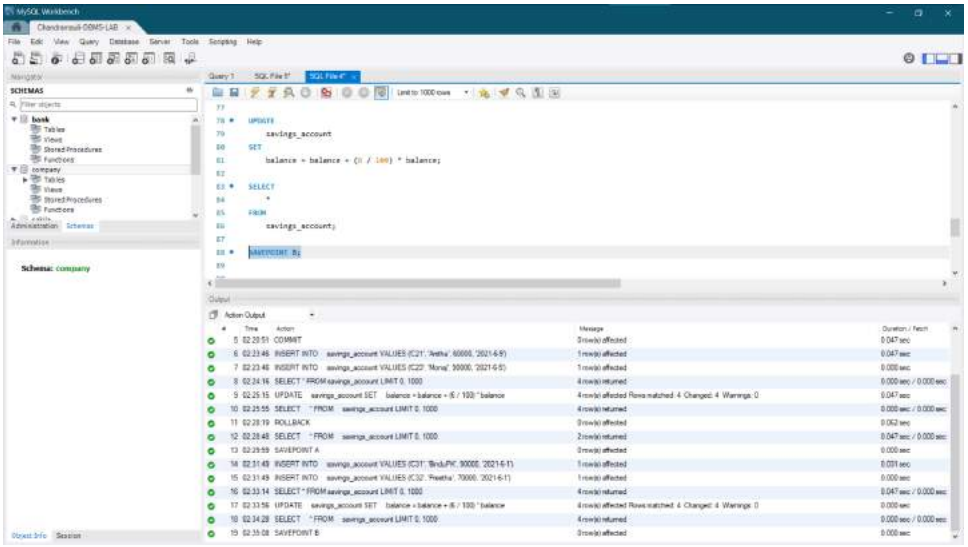
m) Modify the balance amount by adding the interest of 6%.



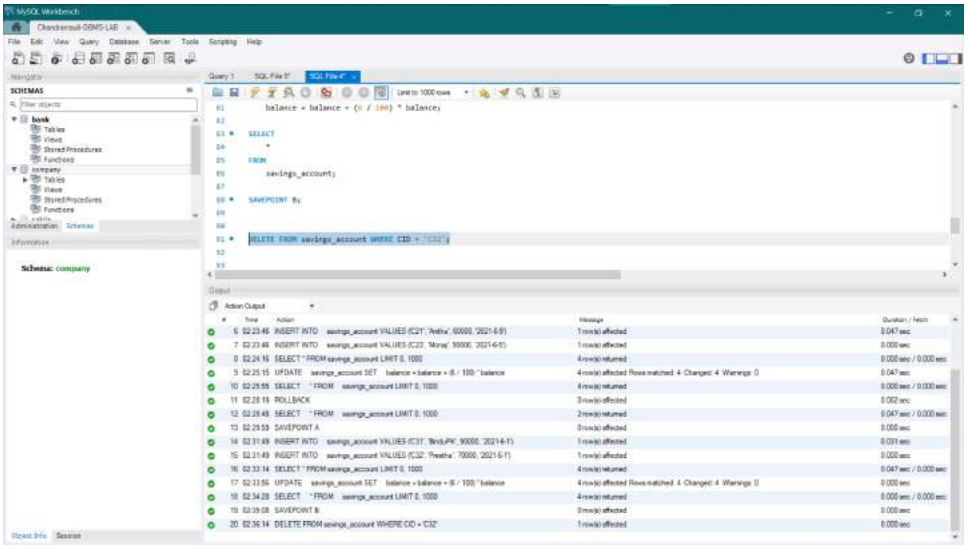
n) Display all the records of ‘savings-account’ table.



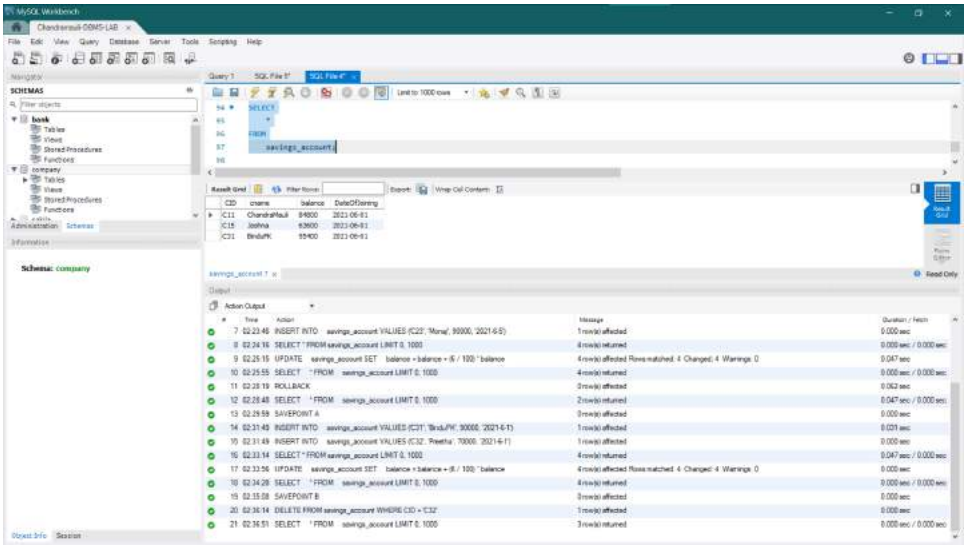
o) Add a marker to the changed state as ‘B’.



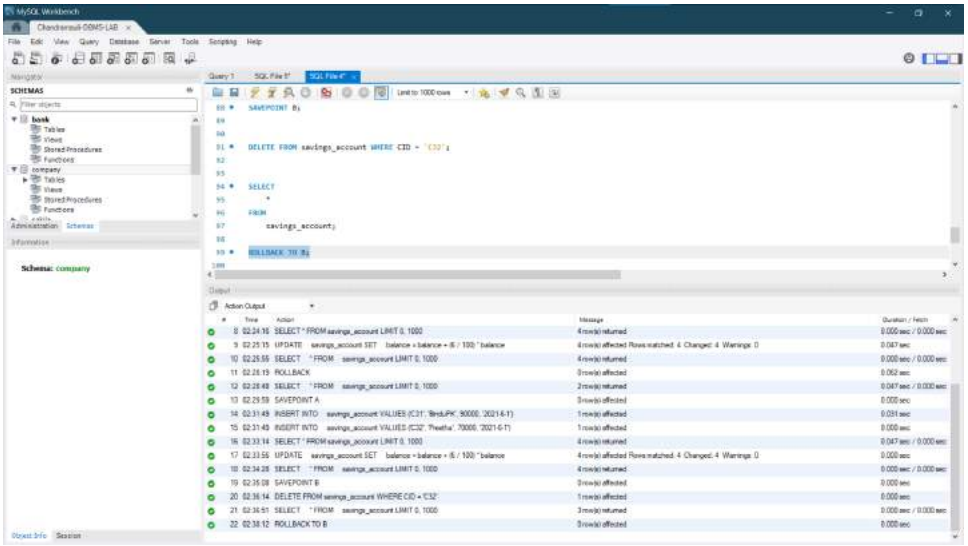
p) Delete one record from the ‘savings-account table.



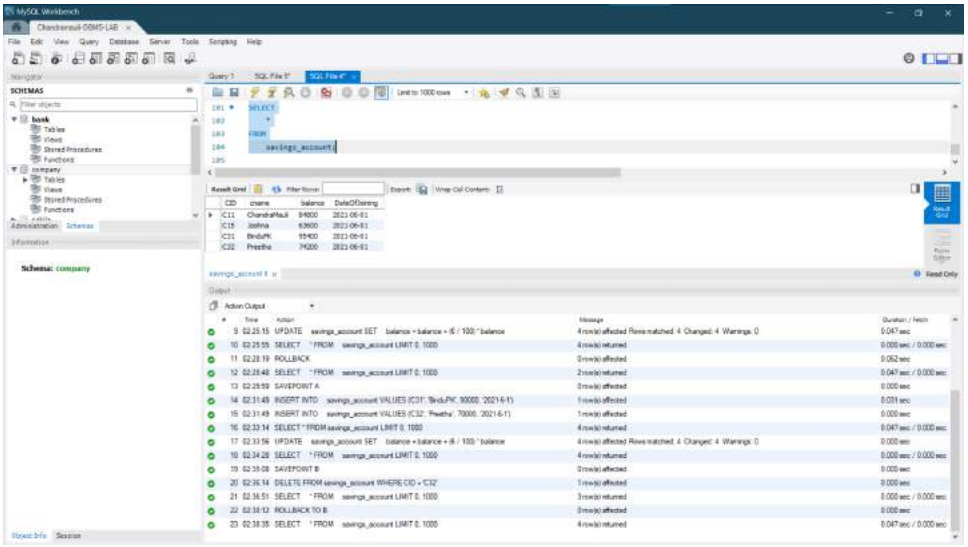
q) Display all the records of ‘savings-account’ table



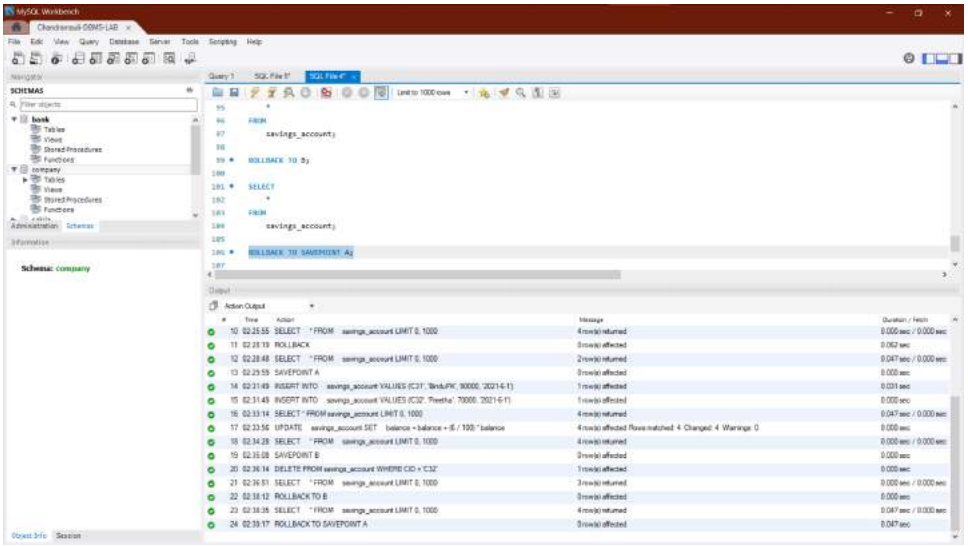
r) Abandon the last deletion (ie, recover the table with deleted row).



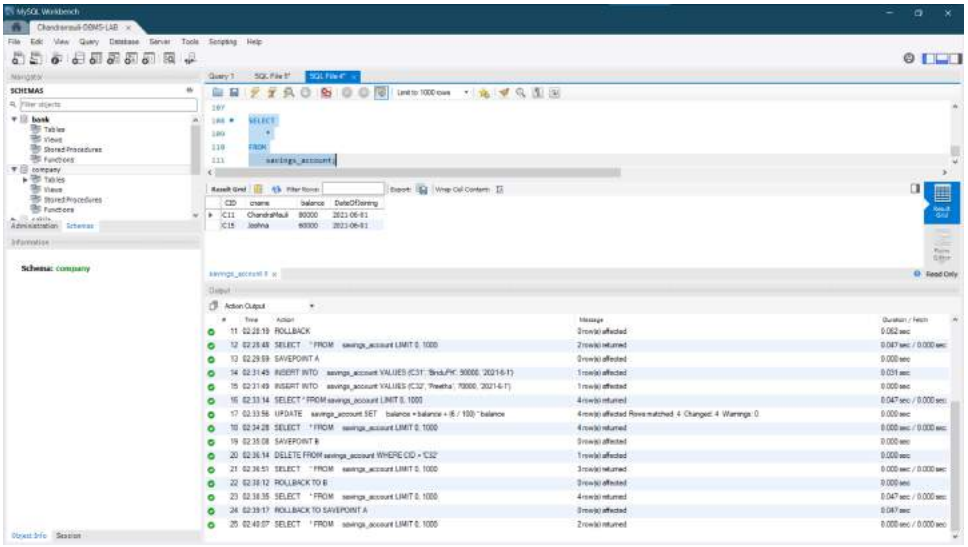
s) Display all the records of ‘savings-account’ table.



t) Abandon to save point/marker ‘A’



u) Display all the records of ‘savings-account’ table.



(Left side of a page)

Name: Suraj Chandramauli **Roll No:**40

Exp. No: 4 **Date: 08-6-2021**

View

Aim: Demonstration of View in SQL **:**

Theory: Create a staff table with arguments staffed, sname, salary, sdept, scategory. Scategory can take 2 values (teaching/ non teaching) only.

Views in SQL are kind of virtual tables. A view also has rows and columns as they are in a real table in the database. We can create a view by selecting fields from one or more tables present in the database. A View can either have all the rows of a table or specific rows based on certain condition.

The uses of Views are:

Restricting data access – Views provide an additional level of table security by restricting access to a predetermined set of rows and columns of a table.

Hiding data complexity – A view can hide the complexity that exists in a multiple table join.

Simplify commands for the user – Views allows the user to select information from multiple tables without requiring the users to actually know how to perform a join.

Store complex queries – Views can be used to store complex queries. Rename Columns – Views can also be used to rename the columns without affecting the base

tables provided the number of columns in view must match the number of columns specified in select statement. Thus, renaming helps to hide the names of the columns of the base tables.

Multiple view facility – Different views can be created on the same table for different users

Syntax for creating a view:

create view view_name as select column1, column2, ... from table_name where condition;

syntax for updating a view:

create or replace view view_name as select column1, column2, ... from table_name where condition;

syntax for deleting a view:

drop view view_name;

example for creating a view:

create view [india customers] as select customername, contactname from customers where country = 'India';

example for updating a view:

create or replace view [india customers] as select customername, contactname, city from customers where country = ‘india’;

example for deleting a view [india]:

drop view ;

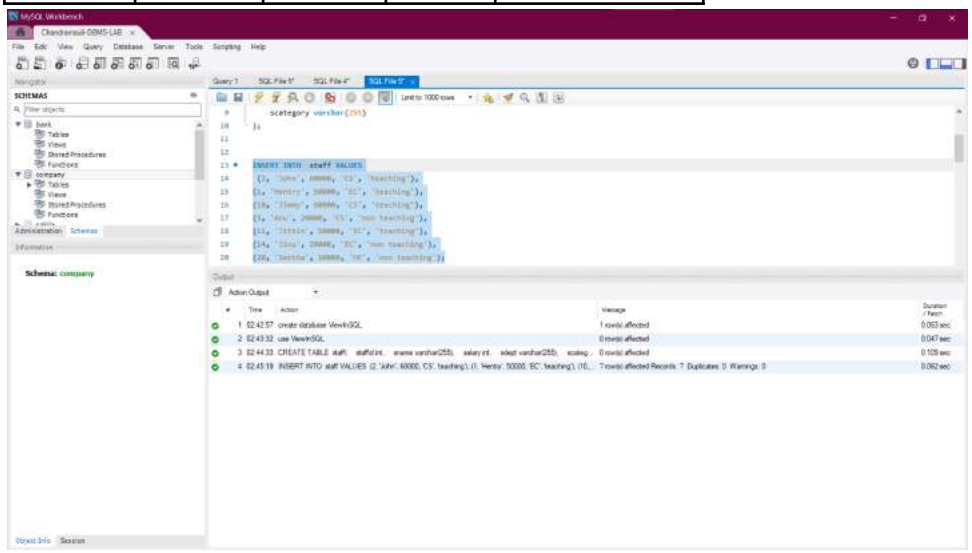
Result: : The view commands are familiarized and output is verified.

Remarks:(To be filled by faculty)

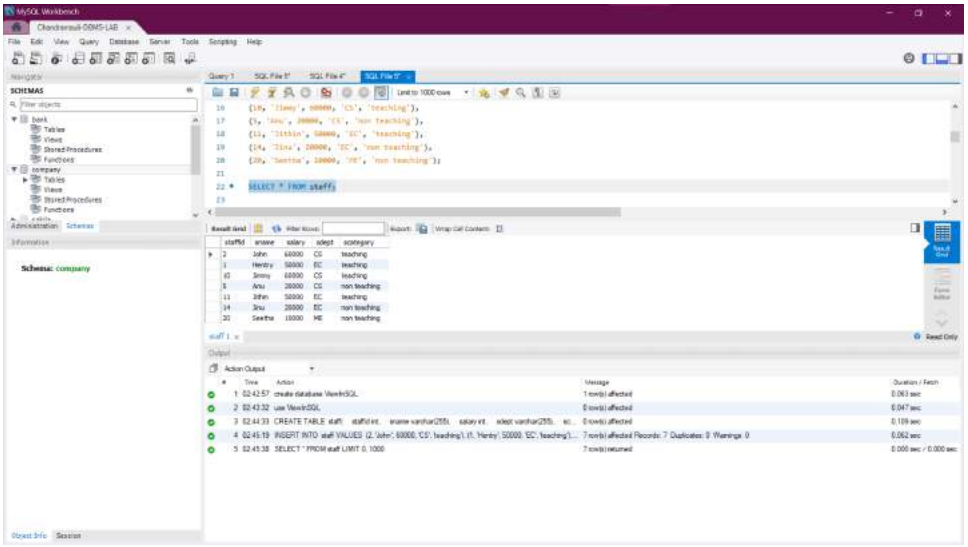
OUTPUT:

a) Insert the following values into staff table.

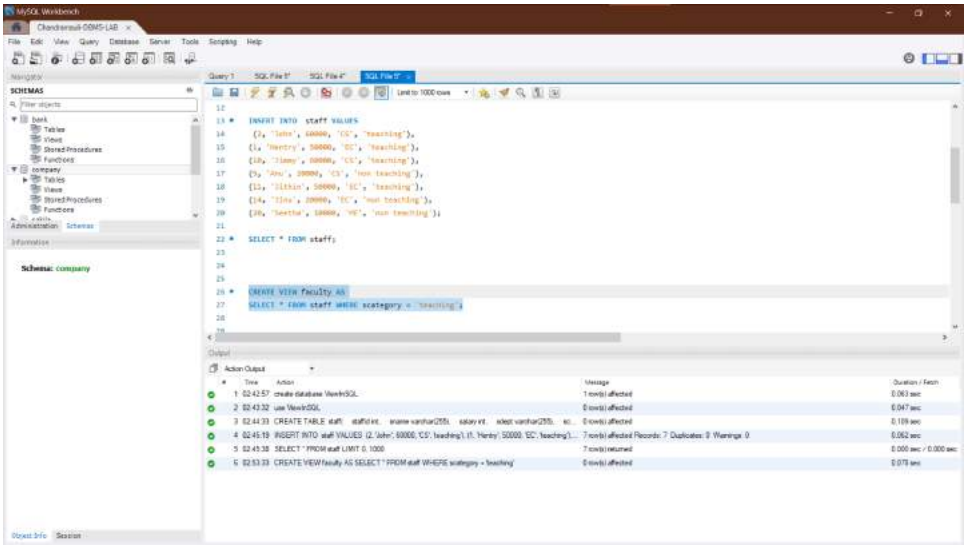
Staffid	sname	salary	sdept	scategory
2	John	60000	CS	teaching
1	Hentry	50000	EC	teaching
10	Jimmy	60000	CS	teaching
5	Anu	20000	CS	non teaching
11	Jithin	50000	EC	teaching
14	Jinu	20000	EC	non teaching
20	Seetha	10000	ME	non teaching



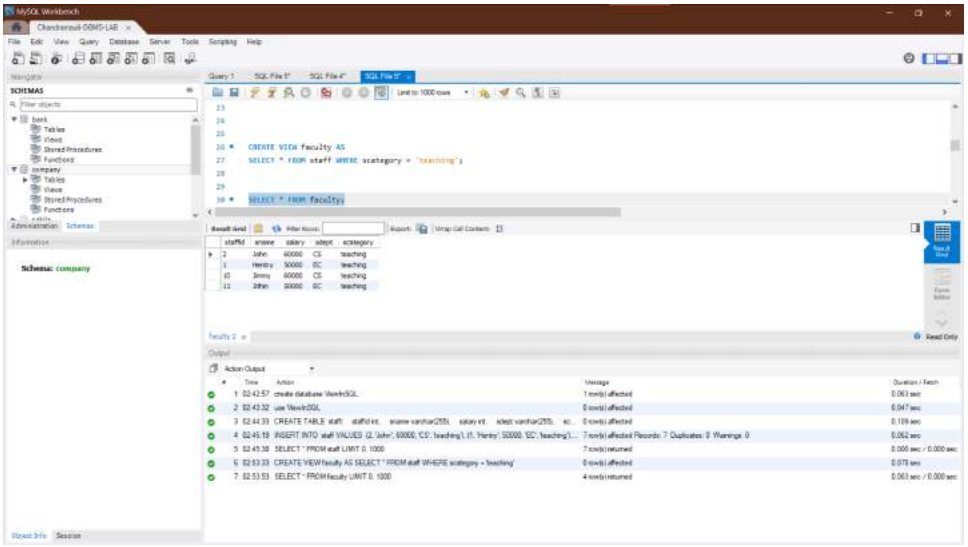
b) Display the details of staff table



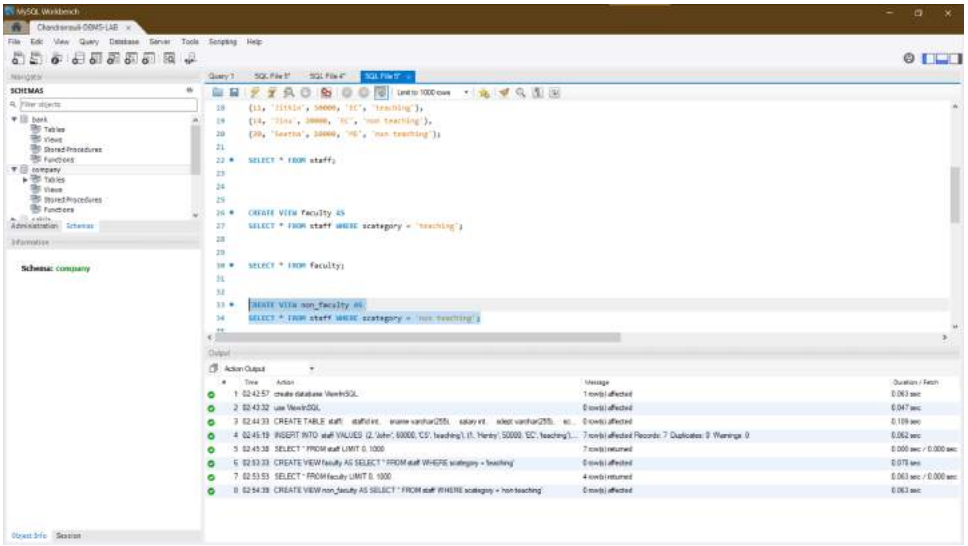
c) Create a view named ‘faculty’ for teaching staff



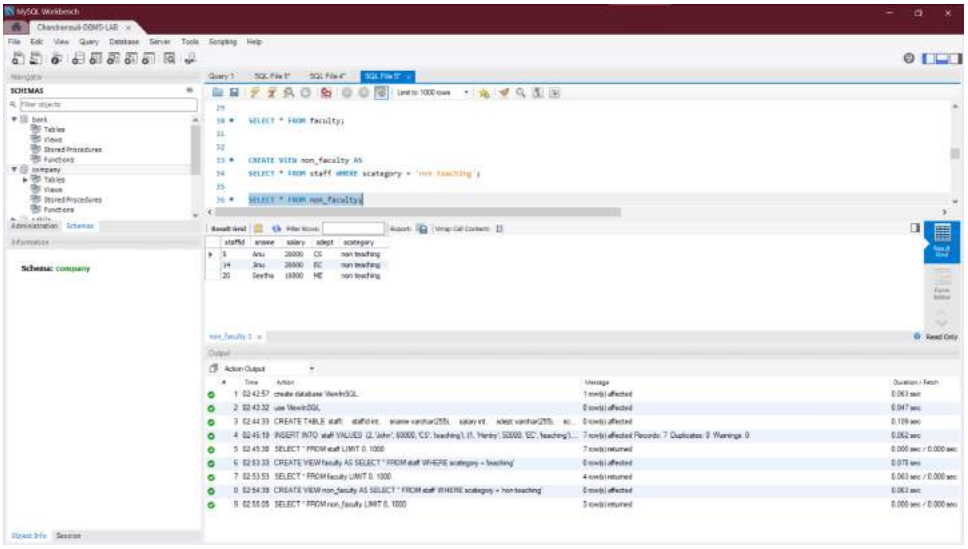
d) Display the contents of ‘faculty’ view.



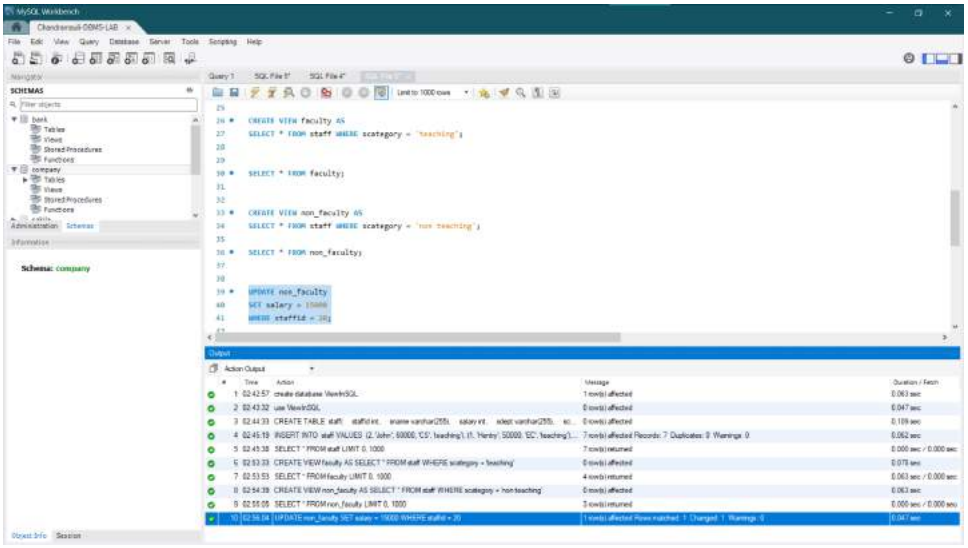
e) Create a view named ‘non-faculty’ for non teaching staff



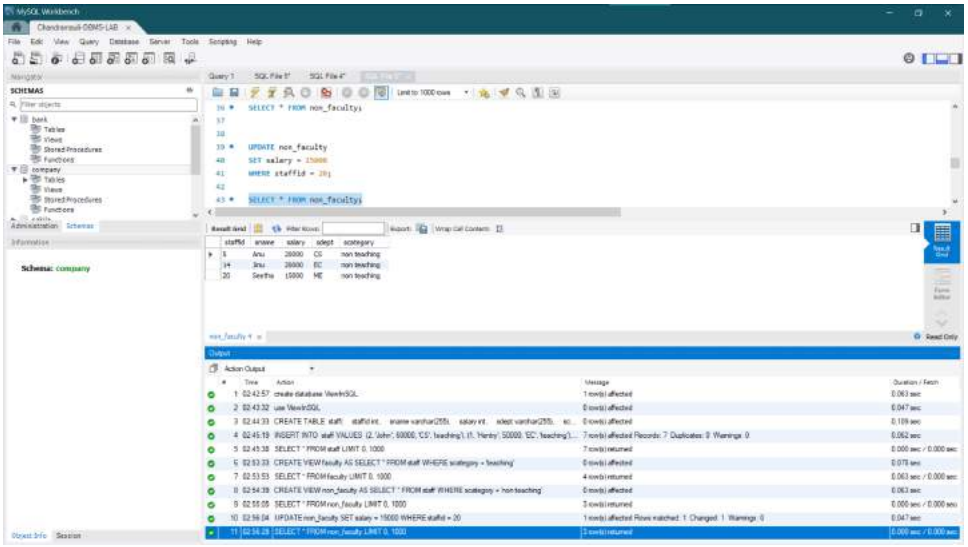
f) Display the contents of ‘non-faculty’ view.



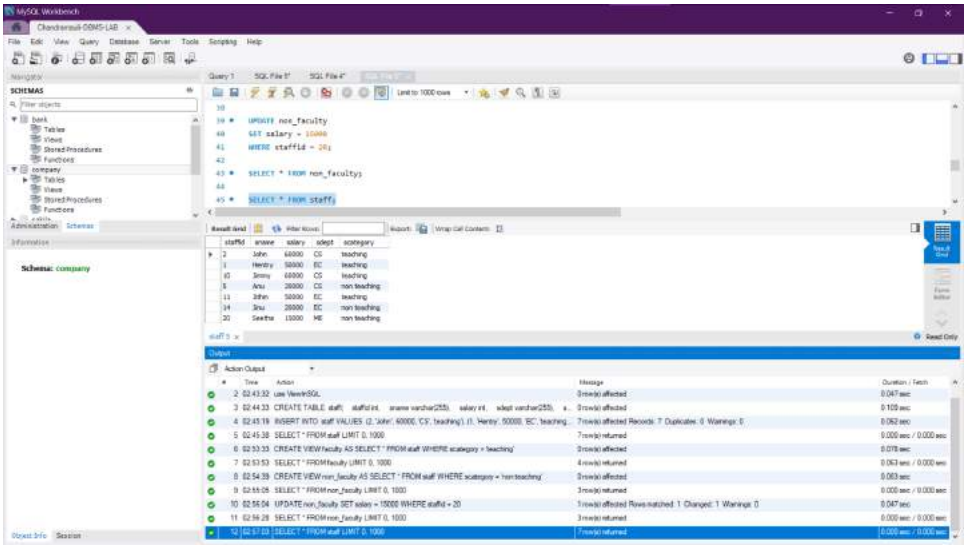
g) Update the salary of non-teaching staff whose staffid=20 to 15000 in corresponding view.



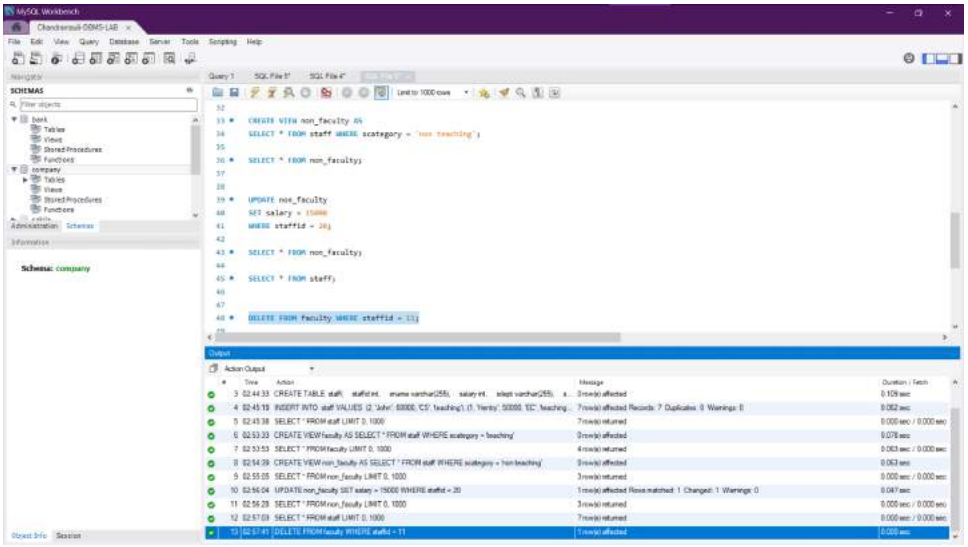
h) Display the contents of ‘non-faculty’ view



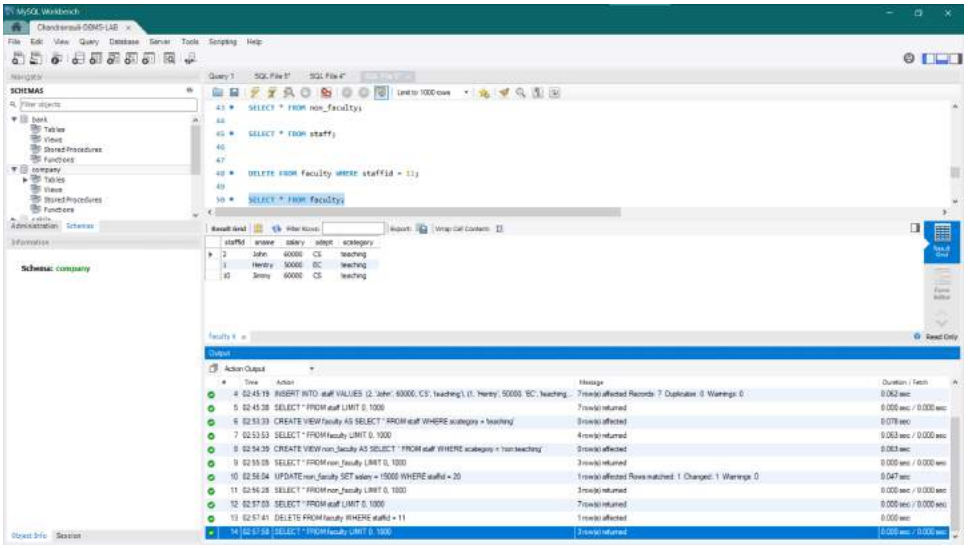
i) Display the contents of staff table.



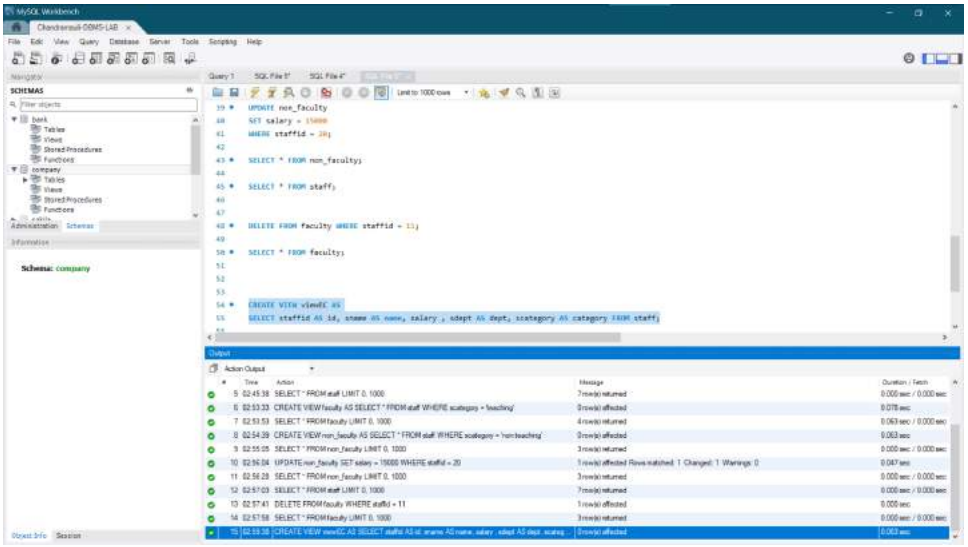
j) Delete the details of staff whose staffid=11 from ‘faculty’ view



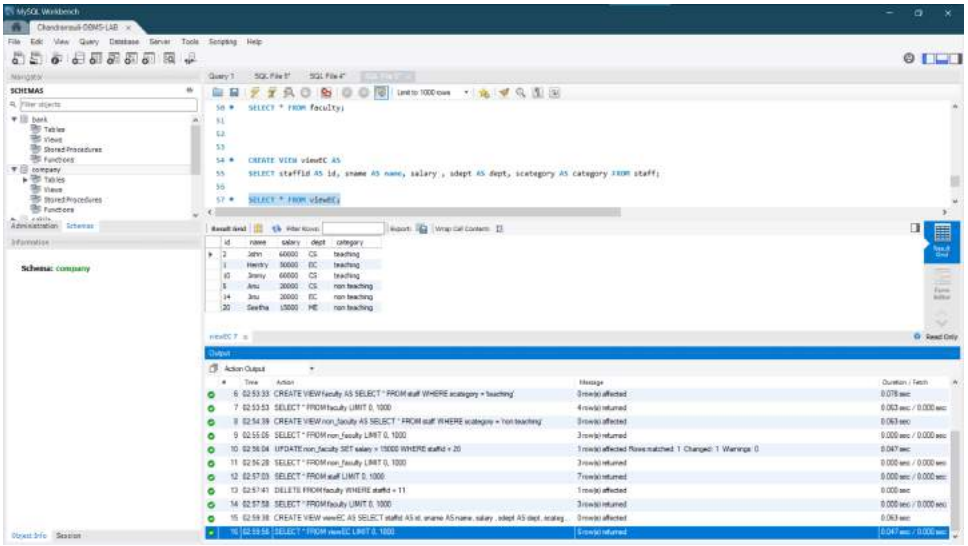
k) Display the contents of ‘faculty’ view.



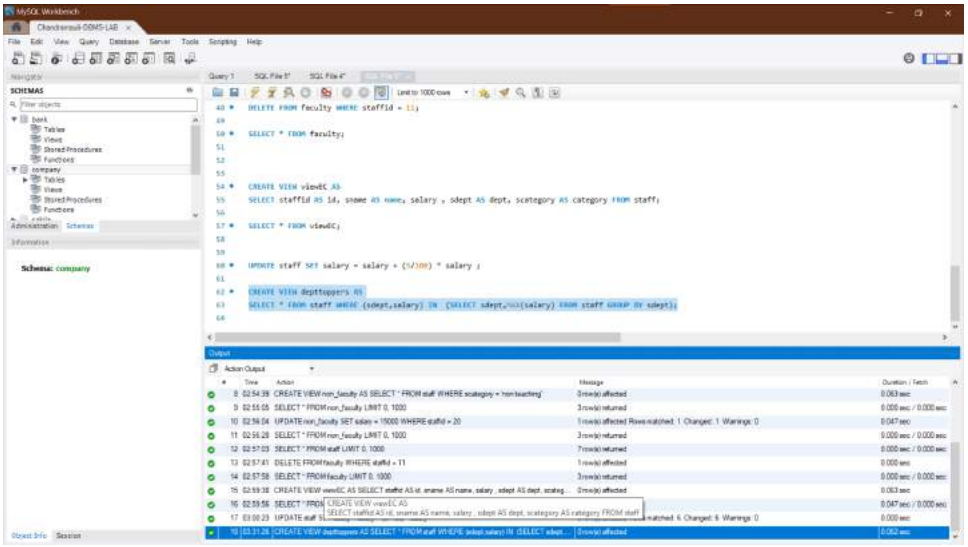
l) Create a view named ‘viewEC’ for staffs in EC department with fieldnames id, name, salary, dept, category respectively.

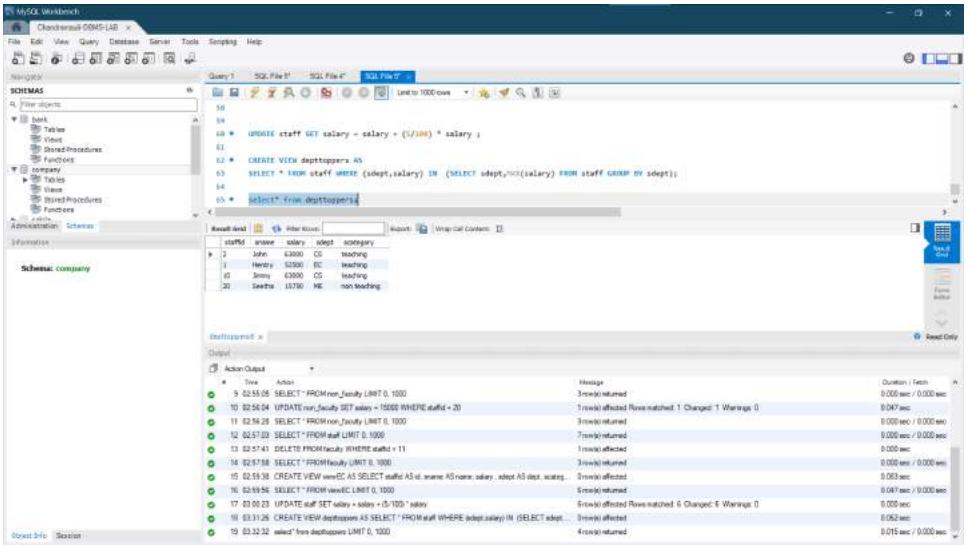


m) Display the contents of ‘viewEC’.



n) Create a view named ‘depttoppers’ for keeping the information of highest salaried staffs in each department.Net salary is calculated by adding the interest of 5%.





(Left side of a page)

Name : Suraj Chandramauli **Roll No:40**

Exp. No:.05 **Date:** 08/06/2021

Complex Queries

Aim: Demonstration of Multiple OR operators and Subqueries:

Theory: A complex query is a parameter query that searches using more than one parameter value i.e. on two or more criteria

It is used for doing multiple AND, OR operations, join operations , set operations and subqueries operations

AND:

This operators displays only those records where both the conditions condition1 and condition2 evaluates to True.

syntax of and is:

select * from table_name where condition1 and condition2 and ...condition; example for and is:

select prod_id, prod_name, prod_price from products where prod_id &alt;= 9 and prod_price <= 90;

OR:

This operators displays the records where either one of the conditions condition1 and condition2 evaluates to True. That is, either condition1 is True or condition2 is True. syntax for or is:

select * from table_name where condition1 or condition2 or... condition; example for or is:

select * from student where name = 'sil' or name = 'trungt';

SUB QUERIES:

A Subquery or Inner query or a Nested query is a query within another SQL query and embedded within the WHERE clause. It is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved. Subqueries can be used with the SELECT, INSERT, UPDATE, and DELETE statements along with the operators like =, , >=, <=, IN, BETWEEN, etc.

· Syntax for Subqueries with SELECT

```
select column_name [, column_name ] from table1 [, table2 ] where column_name operator
(select column_name [, column_name ] from table1 [, table2 ] [where]);
```

Syntax for Subqueries with INSERT:

```
insert into table_name [ (column1 [, column2 ]) ] select [ *|column1 [, column2 ] from table1 [,
table2 ] [ where value operator ];
```

Syntax for Subqueries with UPDATE:

```
UPDATE table SET column_name = new_value [ WHERE OPERATOR [ VALUE ] (SELECT
COLUMN_NAME FROM TABLE_NAME) [ WHERE) ];
```

Syntax for Subqueries with DELETE:

```
DELETE FROM TABLE_NAME [ WHERE OPERATOR [ VALUE ] (SELECT COLUMN_NAME
FROM TABLE_NAME) [ WHERE) ];
```

Example for Subqueries with SELECT:

```
SELECT * FROM CUSTOMERS WHERE ID IN (SELECT ID FROM CUSTOMERS WHERE SALARY
> 5500) ;
```

Example for Subqueries with INSERT:

```
INSERT INTO CUSTO SELECT * FROM CUSTOMERS_LOGS WHERE ID IN (SELECT ID
FROM CUSTO) ;
```

Example for Subqueries with UPDATE:

```
UPDATE CUSTO SET SALARY = SALARY * 0.20 WHERE AGE IN (SELECT AGE FROM
CUSTOMERS_LOGS WHERE AGE >= 37 );
```

Example for Subqueries with DELETE:

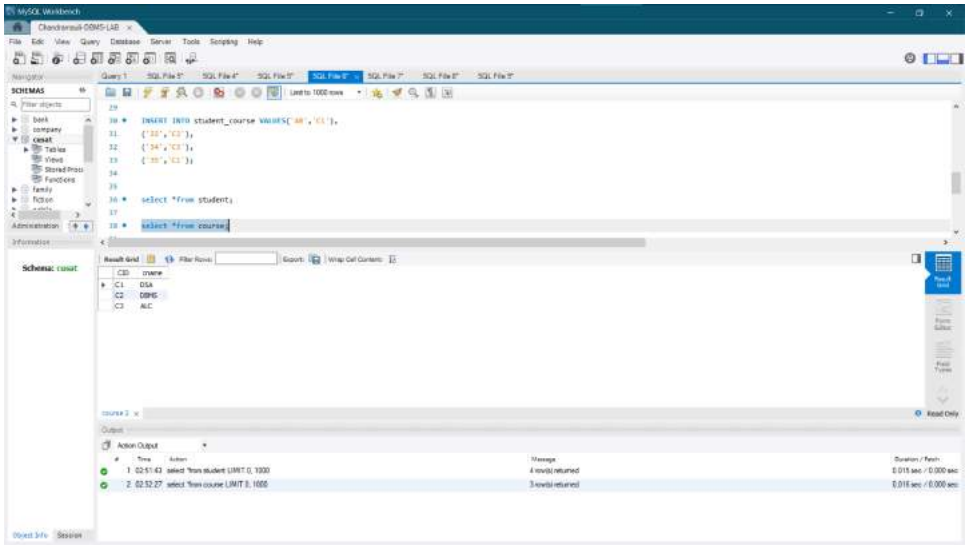
```
DELETE FROM CUSTO WHERE AGE IN (SELECT AGE FROM
CUSTOMERS_LOGS WHERE AGE >= 37 ;
```

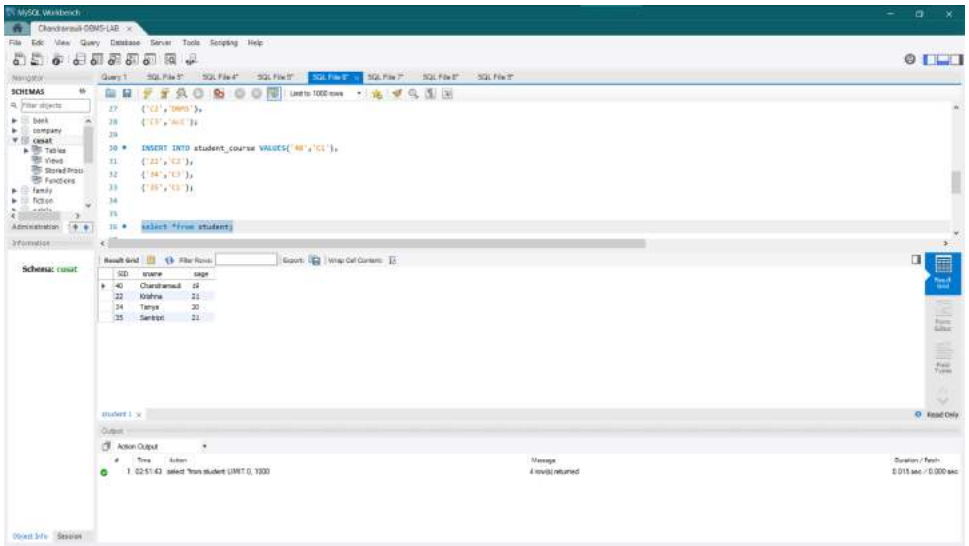
Result: Complex Queries in SQL are familiarized and output is verified.

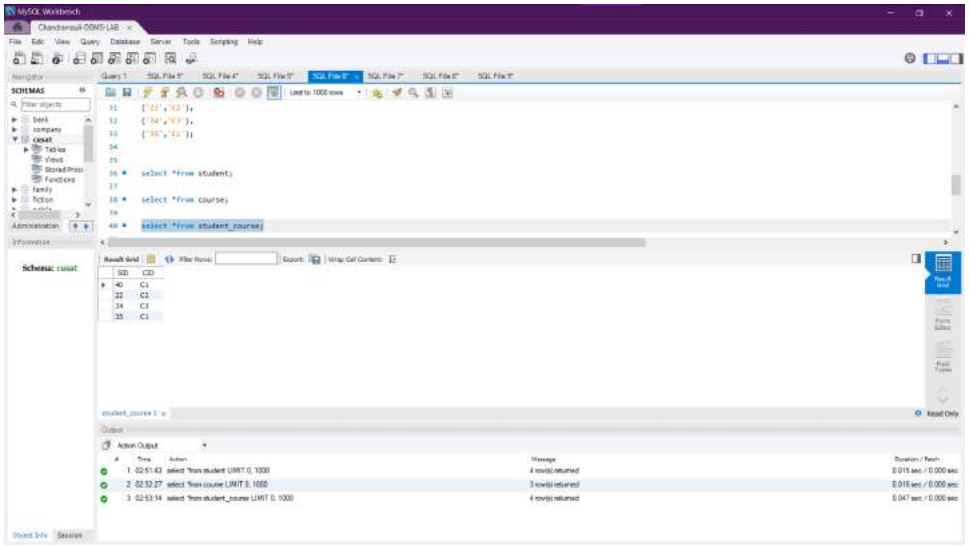
Remarks:(To be filled by faculty)

OUTPUT:

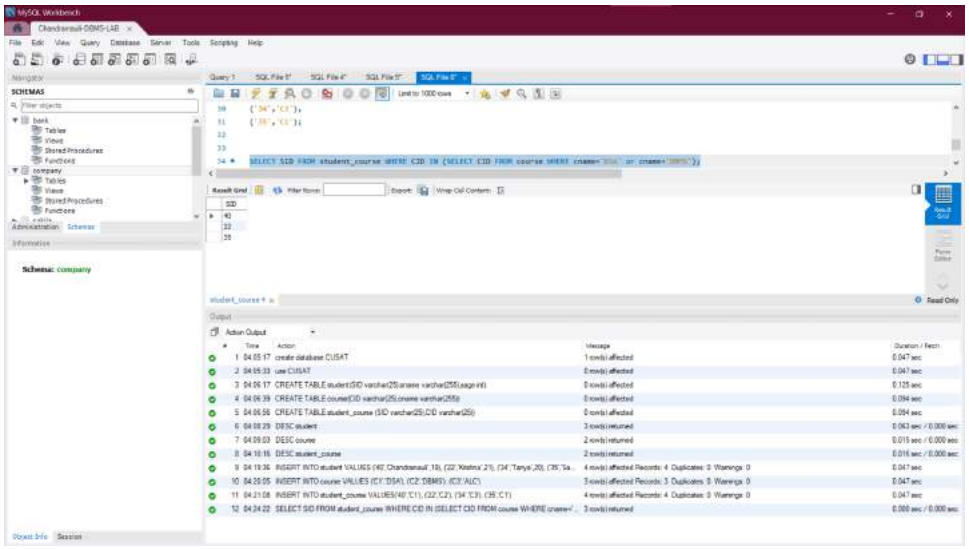
a)Display details of all the 3 tables



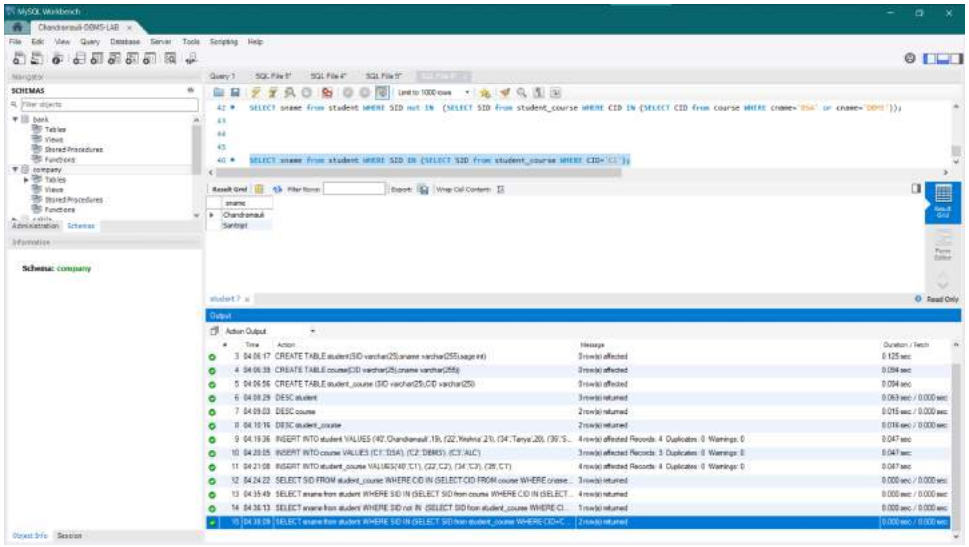




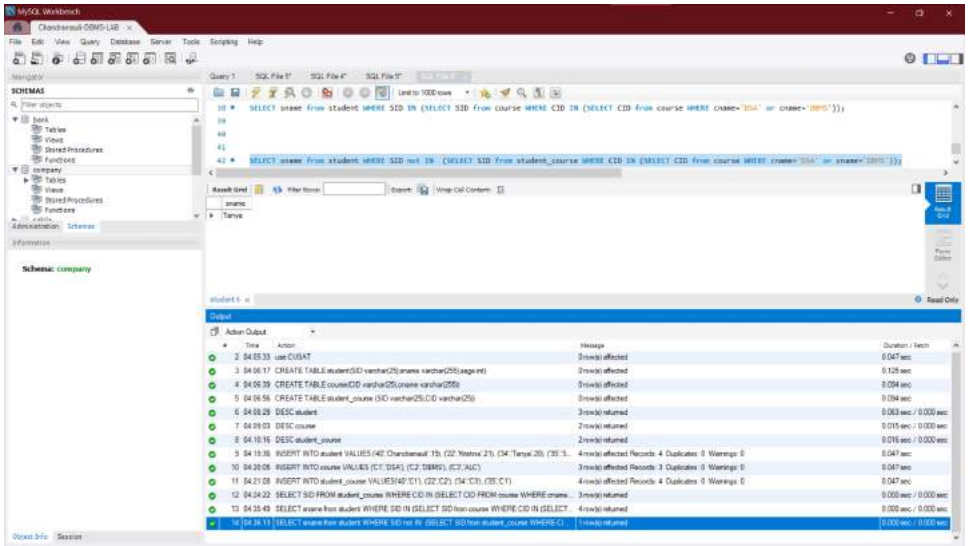
b) Find out student ID (SID) who are enrolled in course name ‘DSA’ or ‘DBMS’



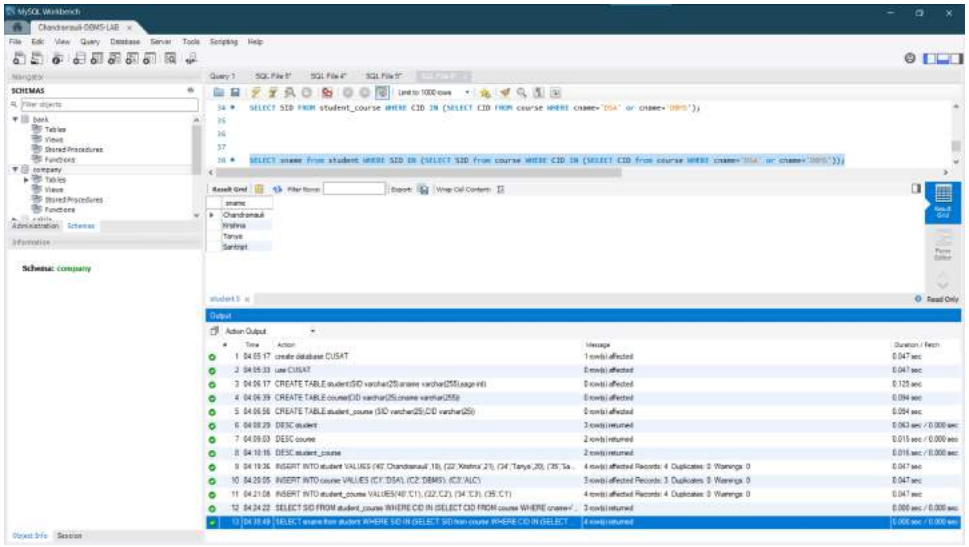
c) Find out names of students who are either enrolled in ‘DSA’ or ‘DBMS’



d) Find out the names of students who are neither enrolled in ‘DSA’ nor in ‘DBMS’



e)Find out the names of students who are enrolled in course ID ‘C1’



(Left side of a page)

Name : Suraj Chandramauli **Roll No:40**

Exp. No:.06 **Date:** 08/06/2021

GROUP BY AND HAVE CLAUSE

Aim: Demonstration of Group By and Have Clause Commands:

Theory:

GROUPING BY: The SQL GROUP BY clause is used in collaboration with the select statement to
arrange identical data into groups.
syntax for group by in sql:
select column1, column2 from table_name where [conditions] group by column1, column2;
example for group by is :
select name, sum(salary) from custo group by name;

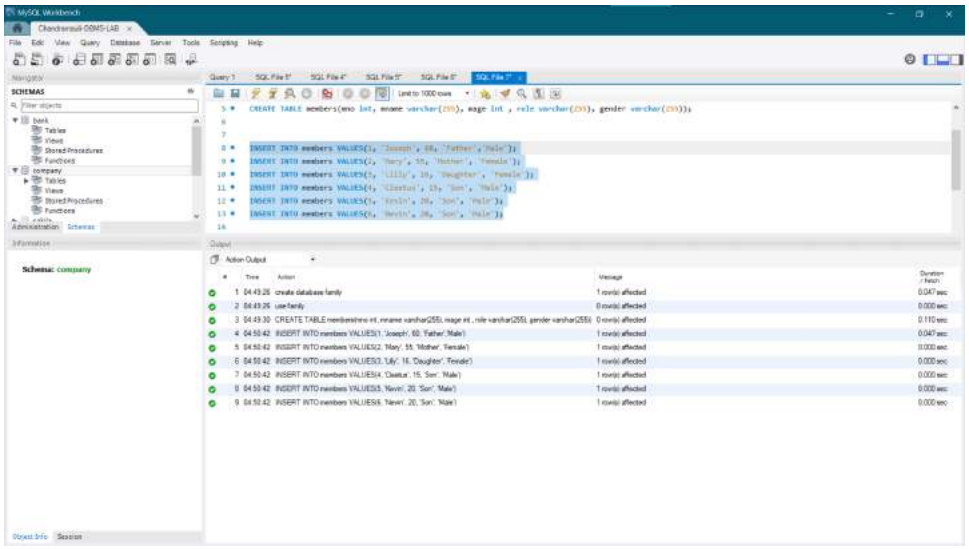
HAVING:
the having clause enables you to specify conditions that filter which group results appear in the results.
syntax for having clause is:
select column1, column2 from table1, table2 where [conditions] group by column1, column2 having [conditions];
example for having clause is:
select cid, name, age, add, salary from custo group by age
having count(age) >= 20;

Result: : GROUP BY and HAVE clause commands are familiarized and output is verified.

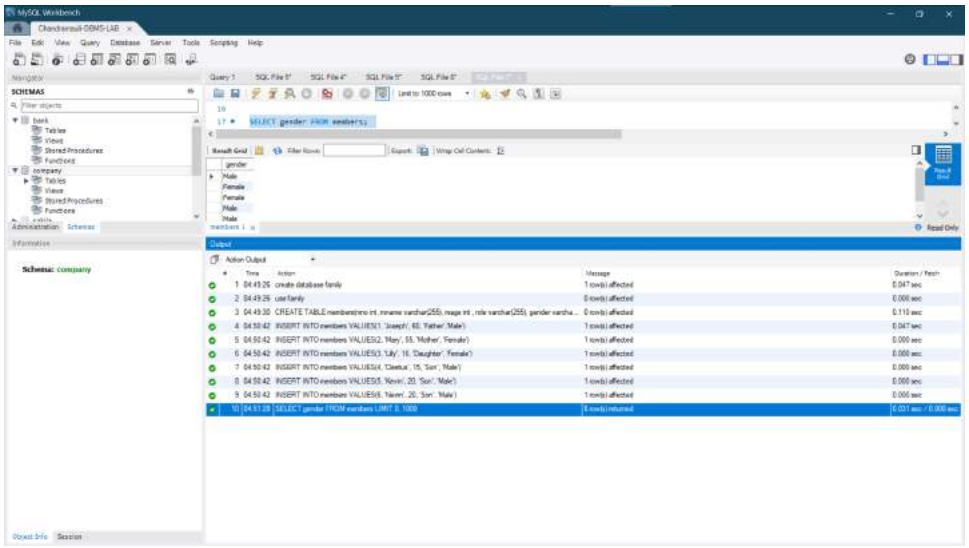
Remarks:(To be filled by faculty)

Output:-

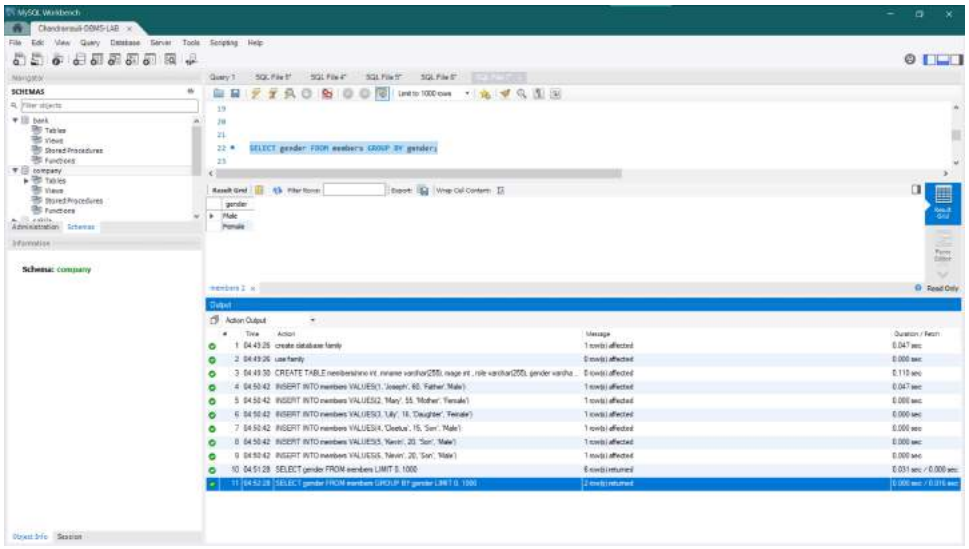
a) Insert following values into ‘members’ table



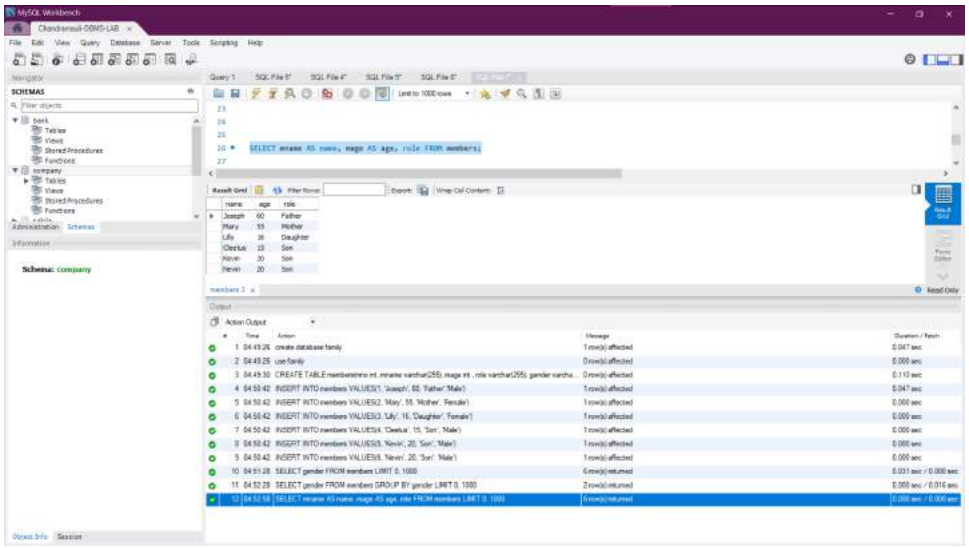
b) Returns all the gender entries from the ‘members’ table



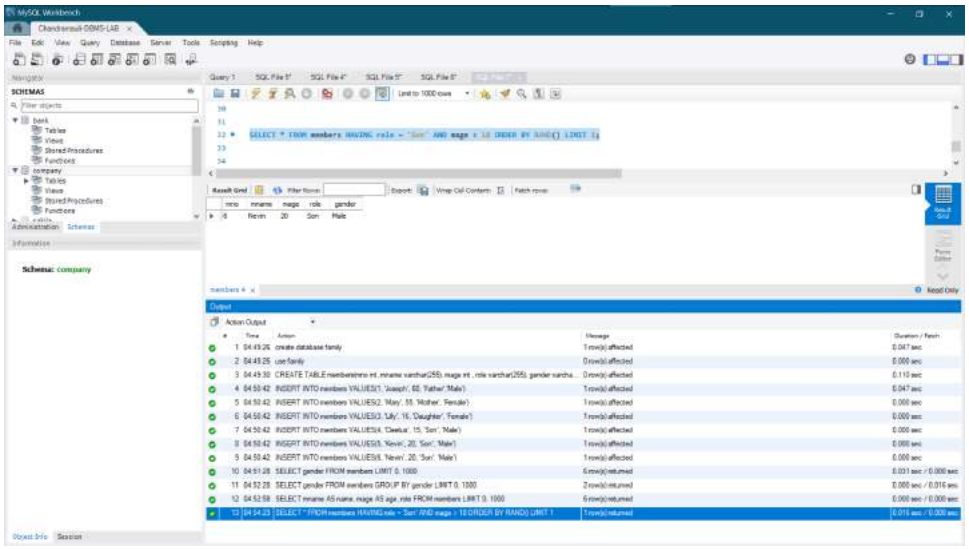
c) Retrieve unique values for genders



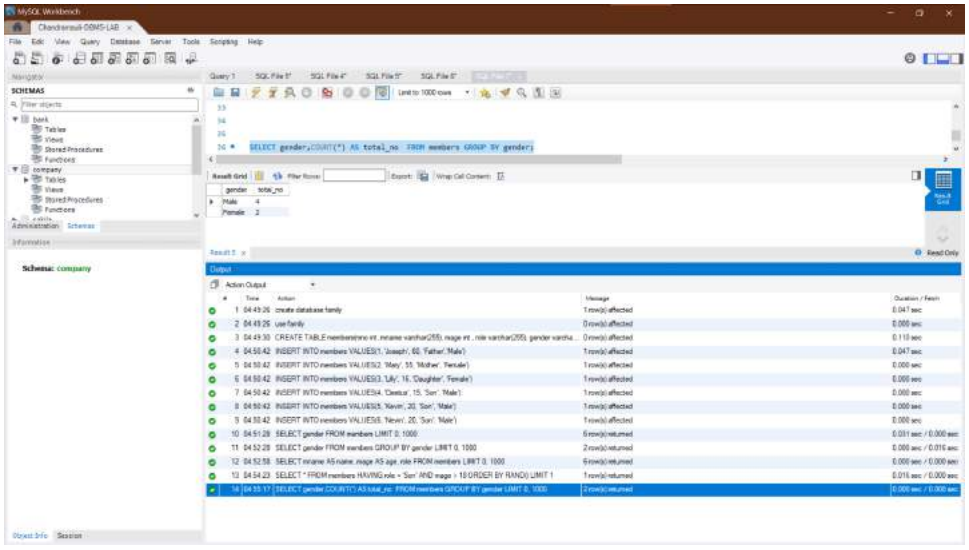
d) Returns all member's name, age, role from 'members' table



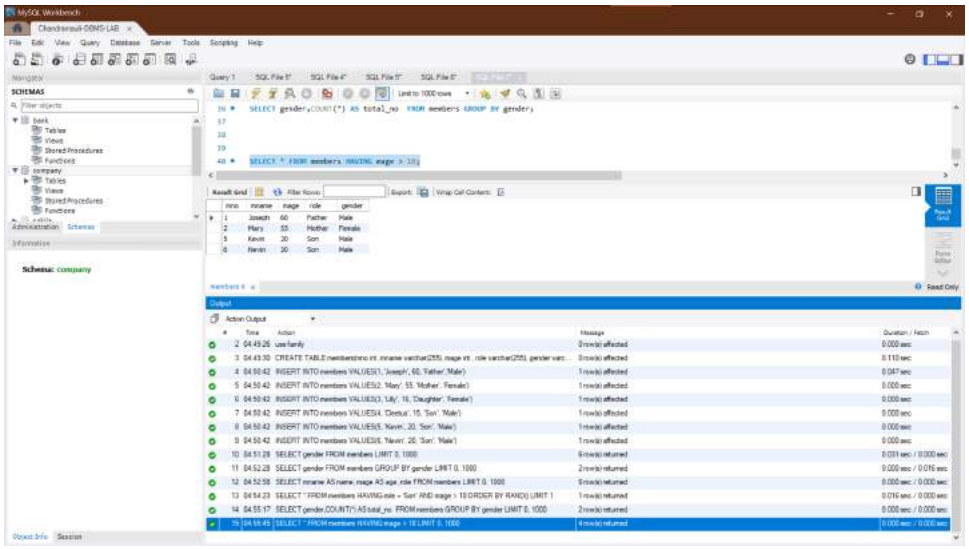
e) Select any one child who is eligible for voting



f) Find out the total no. of males and females in the given family.



g) Select all the eligible candidate's details for voting in the given family



(Left side of a page)

Name: Suraj Chandramauli **Roll No:** 40

Exp. No:7 **Date:** 09/06/2021

JOIN OPERATIONS

Aim: Demonstration of Join Operations:

a) Create student table with following fields

column name	**data type**	**size**
Rollno	Varchar	4

Name	Varchar	15
address	Varchar	15
Phone	Varchar	10
Age	varchar	3

b) Insert the following values into student table

Rollno	**Name**	**address**	**phone**	**Age**
1	ram	Delhi	9961253564	18
2	Ramesh	Gurgaon	9962363564	18
3	Sujit	Rohtak	9961253222	20
4	Suresh	delhi	9961663564	18

c) Create studentcourse table with following fields

Column name	**Data type**	**Size**
Courseid	number	3
rollno	Number	3

d) Insert the following values into studentcourse table

courseid	**Rollno**

1	1
2	2
2	3
3	4

e) Display all the values of employee table.

f) Display all the values of customer table.

g) select NAME and Age from Student table and COURSEID from StudentCourse table. (cross join)

h) each row of the student table is joined with itself and all other rows depending on some conditions(eg: a.ROLL_NO < b.ROLL_NO). (self join)

o/p for given eg:

Roll no	Name
1	Ramesh
1	Sujit
2	Sujit
1	Suresh
2	Suresh
3	Suresh

i) Show the names and age of students enrolled in different courses. (equi join)

j) Perform natural join on ‘student’ and ‘studentcourse’ table.

k) Perform left join on ‘student’ and ‘studentcourse’ table.

l) Perform right join on ‘student’ and ‘studentcourse’ table.
m)Perform full outer join on ‘student’ and ‘studentcourse’ table

Theory:

A SQL Join statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are: INNER JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN.

INNER JOIN:

· The INNER JOIN keyword selects all rows from both the tables as long as the condition satisfies. This keyword will create the result-set by combining all rows from both the tables where the condition satisfies i.e value of the common field will be same.

· Syntax for INNER JOIN is:

SELECT table1.column1,table1.column2,table2.column1,.... FROM table1 INNER JOIN table2 ON table1.matching_column = table2.matching_column;

· Example of INNER JOIN is:

SELECT StudentCourse.COURSE_ID, Student.NAME, Student.AGE FROM Student INNER JOIN StudentCourse ON Student.ROLL_NO = StudentCourse.ROLL_NO;

LEFT JOIN:

· This join returns all the rows of the table on the left side of the join and matching rows for the table on the right side of join. The rows for which there is no matching row on right side, the result-set will contain null. LEFT JOIN is also known as LEFT OUTER JOIN. · Syntax for LEFT JOIN is:

SELECT table1.column1,table1.column2,table2.column1,.... FROM table1 LEFT JOIN table2 ON table1.matching_column = table2.matching_column;

· Example of LEFT JOIN is:

SELECT Student.NAME,StudentCourse.COURSE_ID FROM Student LEFT JOIN StudentCourse ON StudentCourse.ROLL_NO = Student.ROLL_NO;

RIGHT JOIN:

· RIGHT JOIN is similar to LEFT JOIN. This join returns all the rows of the table on the right side of the join and matching rows for the table on the left side of join. The rows for which there is no matching row on left side, the result-set will contain null. RIGHT JOIN is also known as RIGHT OUTER JOIN.

· Syntax for RIGHT JOIN is:

SELECT table1.column1,table1.column2,table2.column1,.... FROM table1 RIGHT JOIN table2 ON table1.matching_column = table2.matching_column;

· Example of RIGHT JOIN is:

SELECT Student.NAME,StudentCourse.COURSE_ID FROM Student RIGHT JOIN StudentCourse ON StudentCourse.ROLL_NO = Student.ROLL_NO;

FULL JOIN:

· FULL JOIN creates the result-set by combining result of both LEFT JOIN and RIGHT JOIN. The result-set will contain all the rows from both the tables. The rows for which there is no matching, the result-set will contain NULL values

· Syntax for FULL JOIN is:

SELECT table1.column1,table1.column2,table2.column1,.... FROM table1 FULL JOIN table2 ON table1.matching_column = table2.matching_column;

· Example of FULL JOIN is:

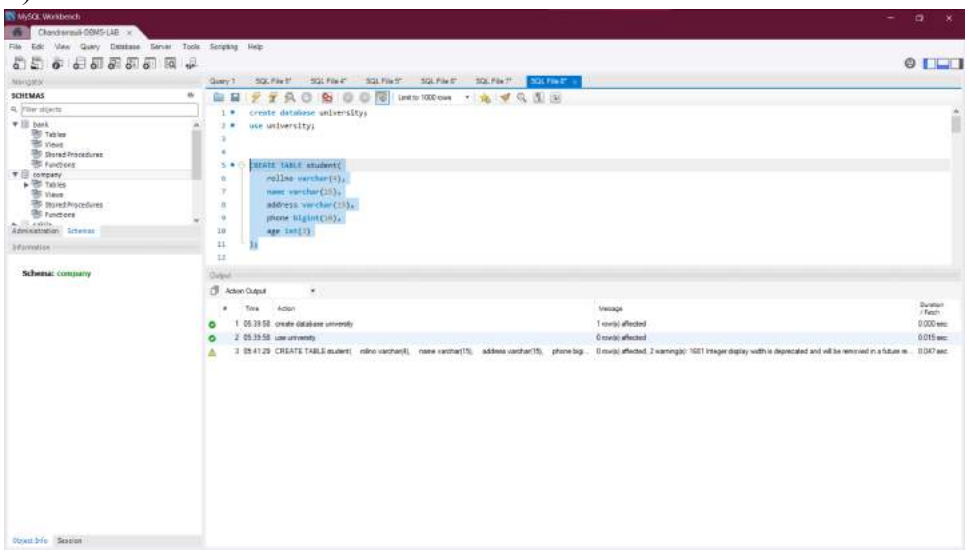
SELECT Student.NAME,StudentCourse.COURSE_ID FROM Student FULL JOIN StudentCourse ON StudentCourse.ROLL_NO = Student.ROLL_NO;

Result: Join Operations are familiarized and output is verified.

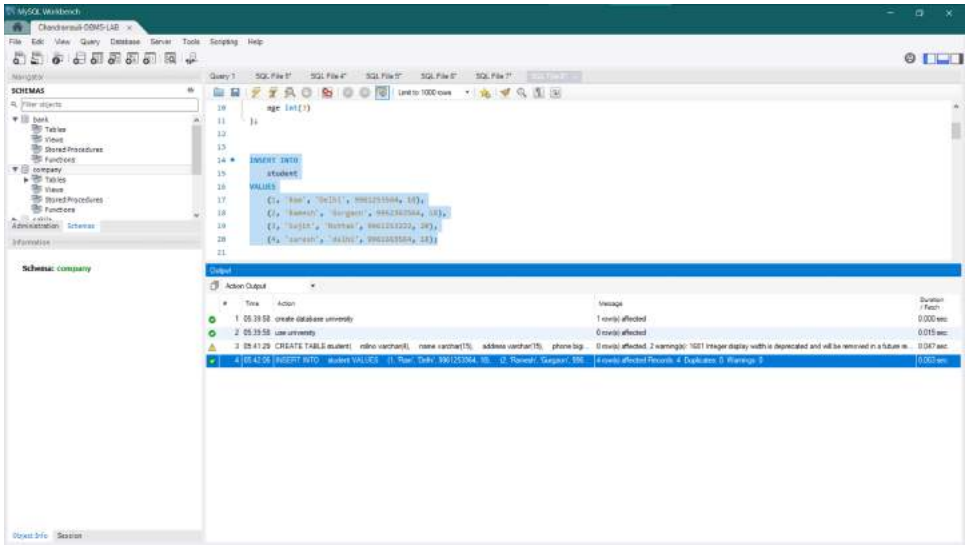
Remarks:(To be filled by faculty)

OUTPUT:

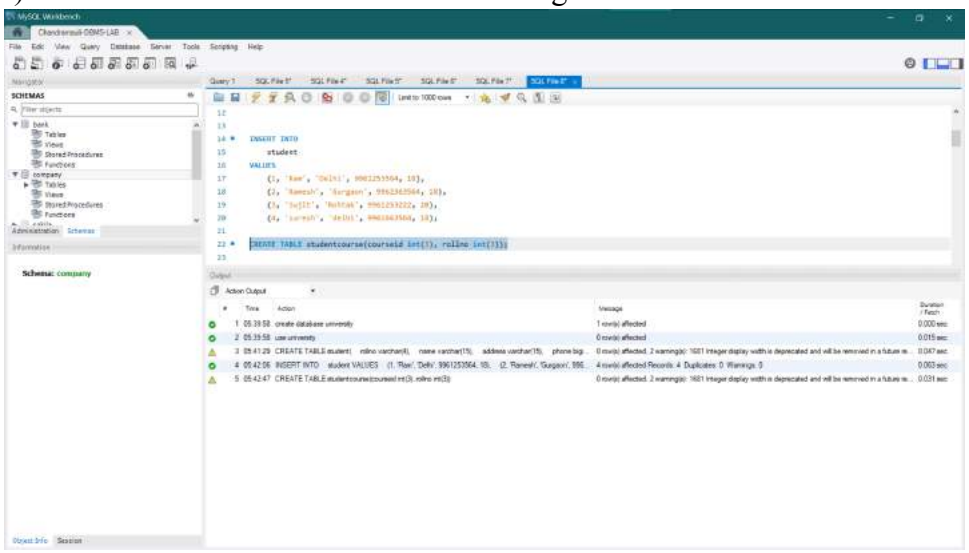
a) Create student table with fields



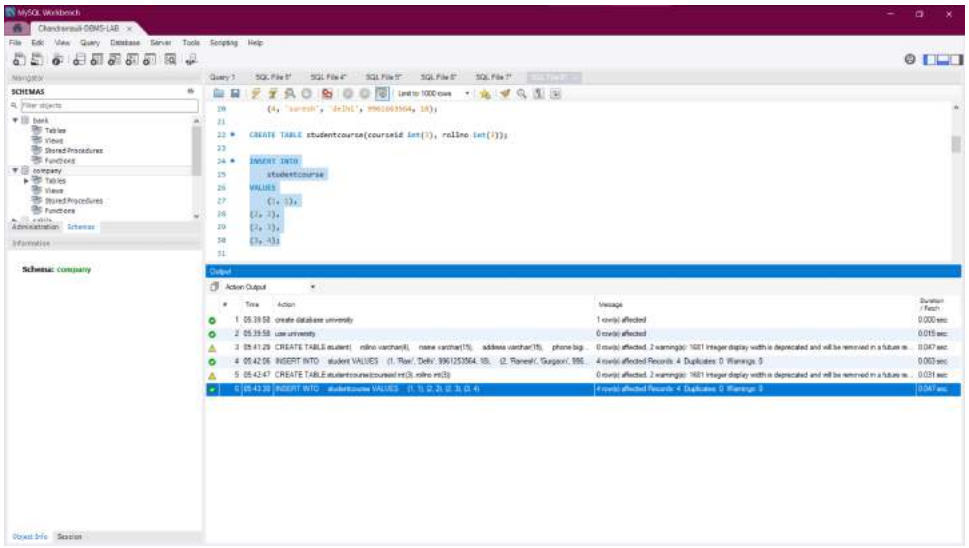
b) Insert the following values into student table



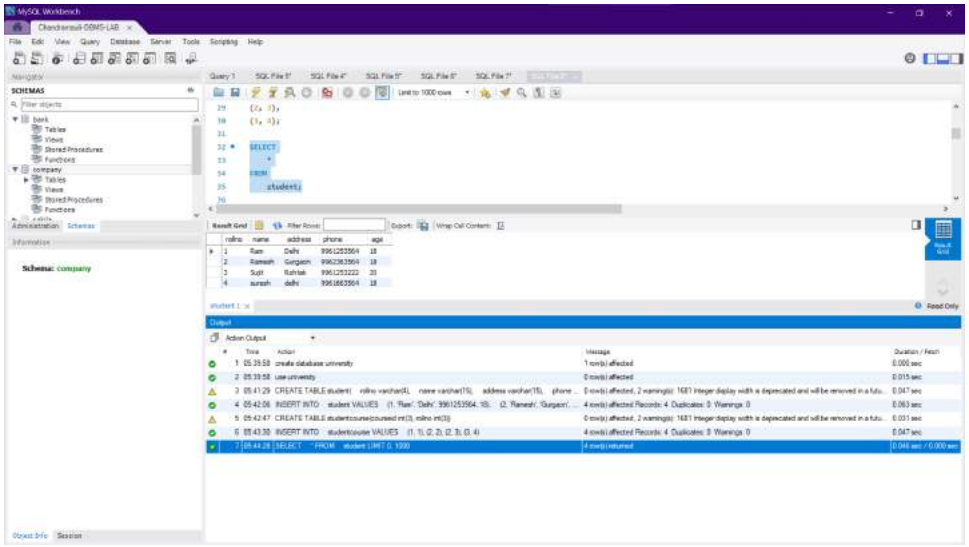
c) Create studentcourse table with following fields



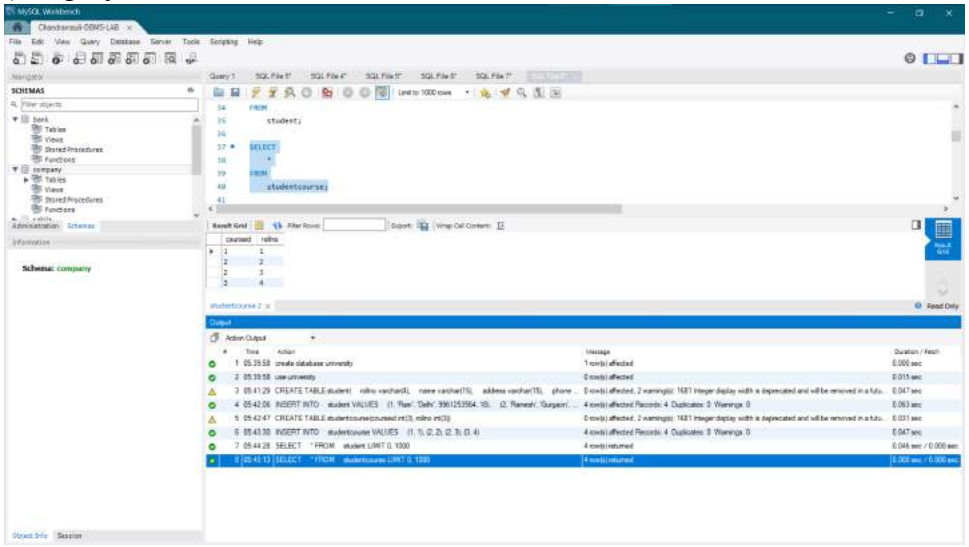
d) Insert the following values into studentcourse table



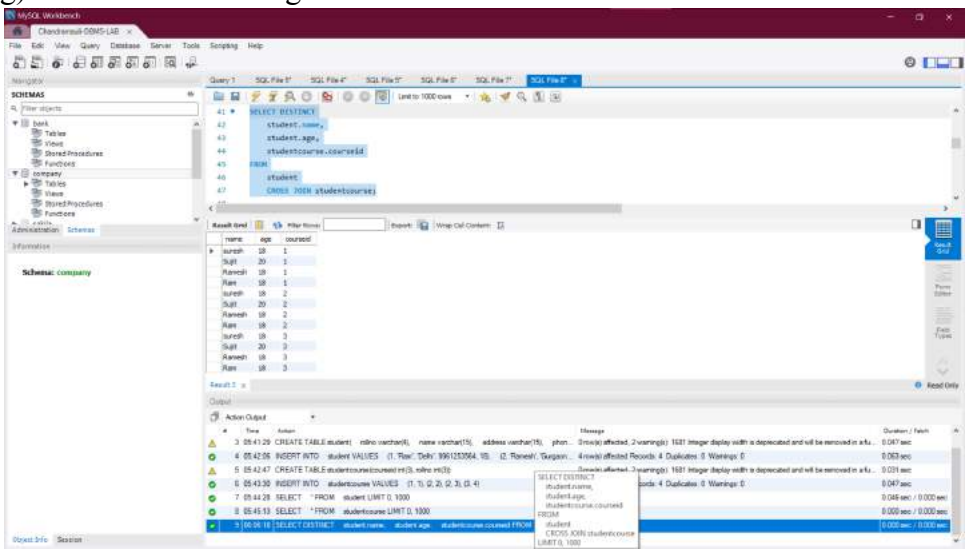
e)Display all the values of employee table.



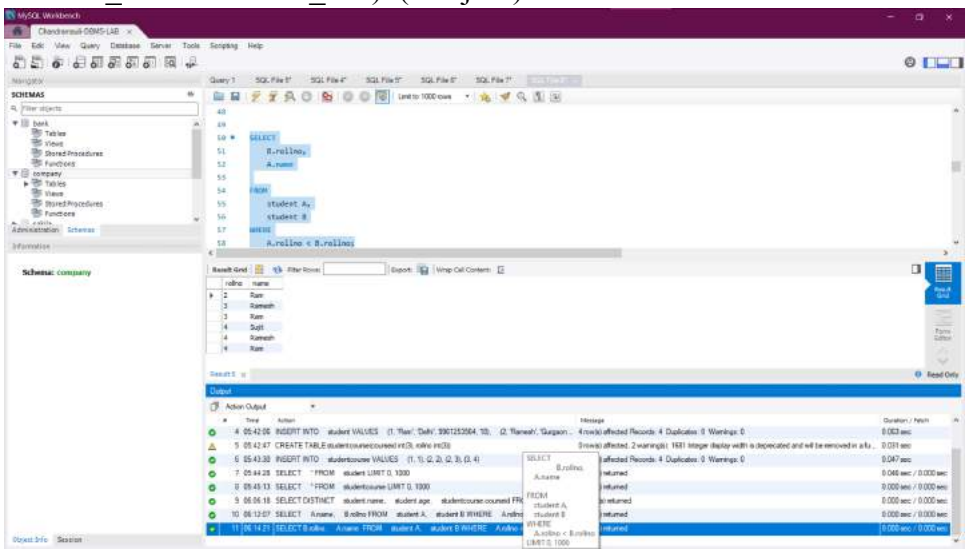
f)Display all the values of customer table



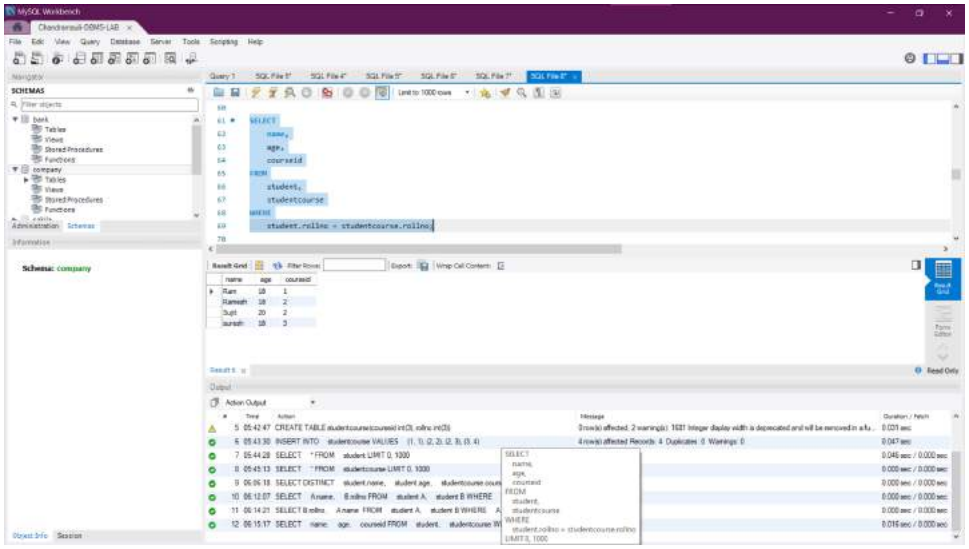
g)select NAME and Age from Student table and COURSEID from StudentCourse table. (cross join)



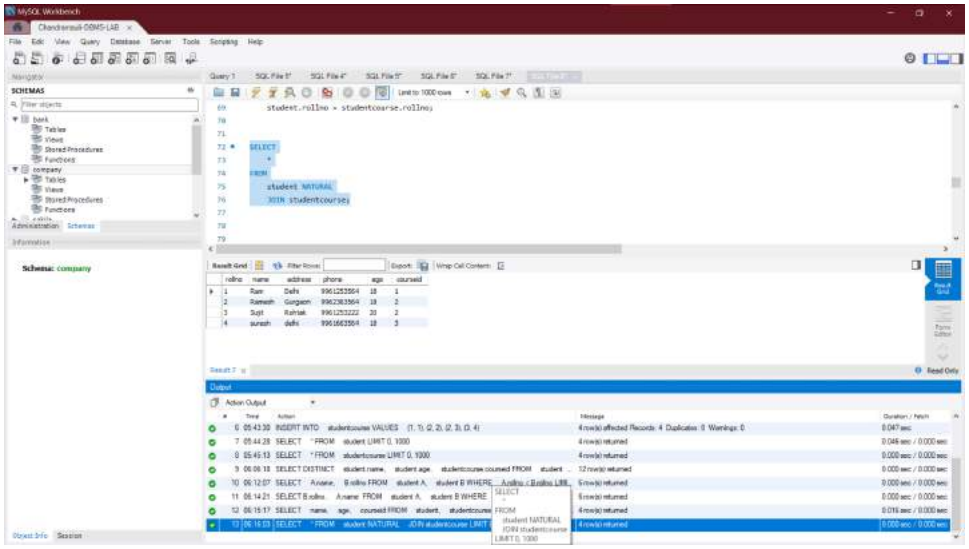
h)each row of the student table is joined with itself and all other rows depending on some conditions(eg: a.ROLL_NO < b.ROLL_NO). (self join)



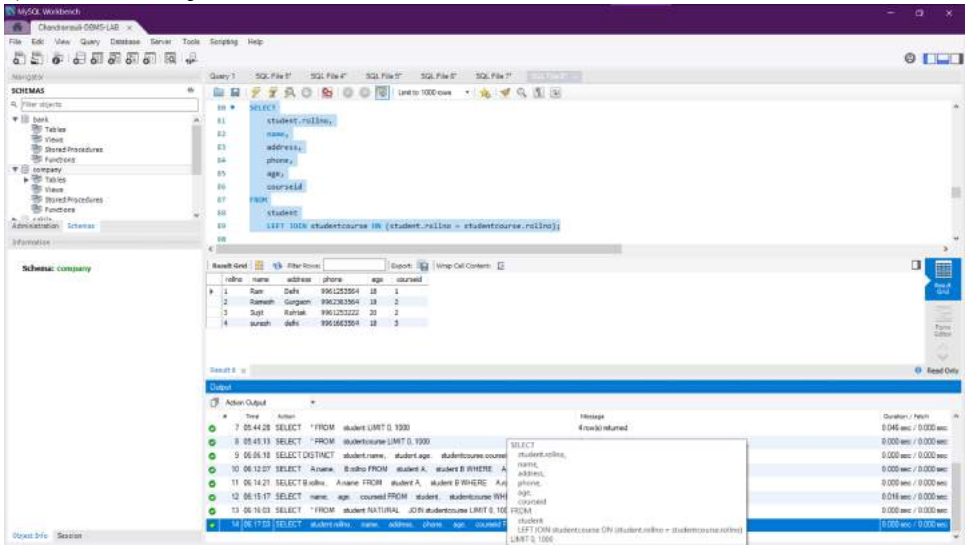
i) Show the names and age of students enrolled in different courses. (equi join)



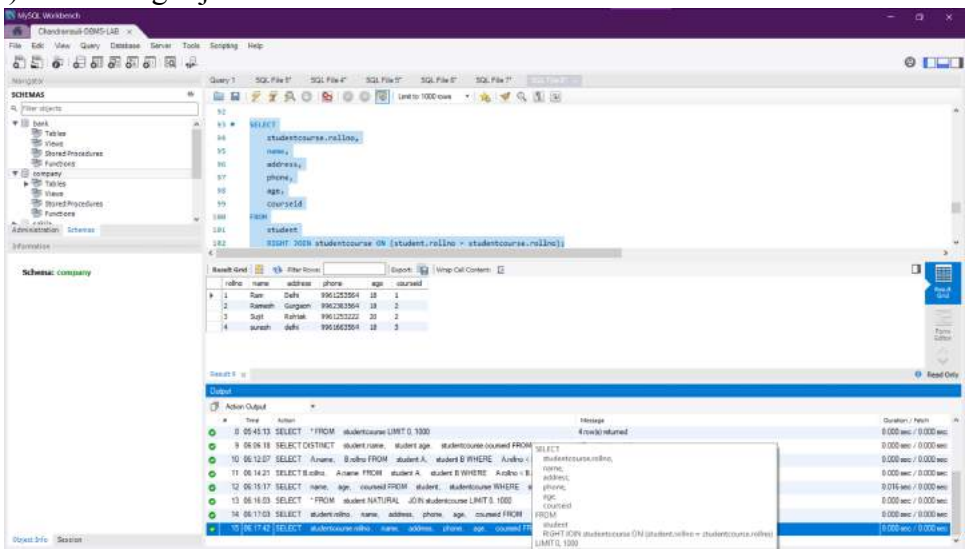
j)Perform natural join on ‘student’ and ‘studentcourse’ table.



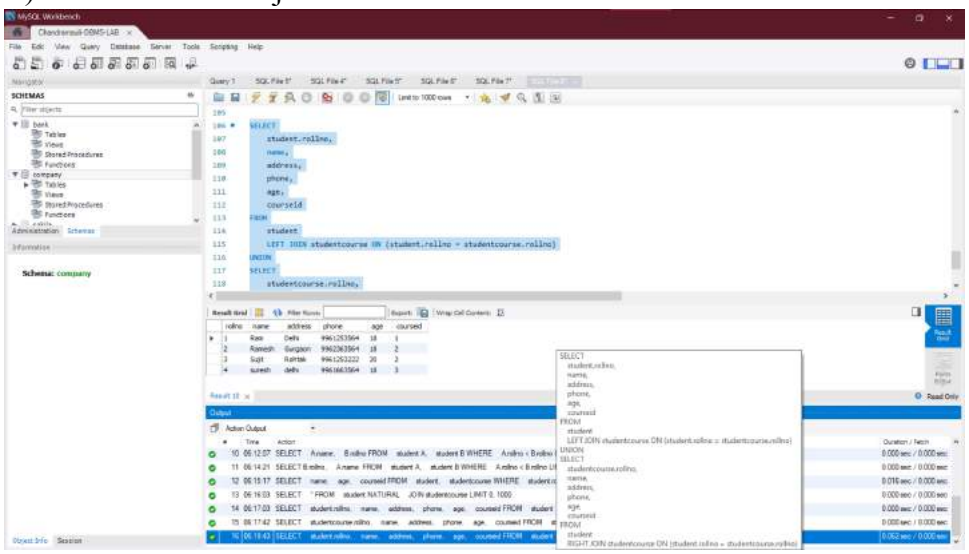
k)Perform left join on ‘student’ and ‘studentcourse’ table



l)Perform right join on ‘student’ and ‘studentcourse’ table.



m)Perform full outer join on ‘student’ and ‘studentcourse’ table



(Left side of a page)

Name: Suraj Chandramauli **Roll No: 40**

Exp. No: 8 **Date:** 09/06/2021

SET OPERATIONS

Aim: Demonstration of Set Operations:

Based on the following given tables, write query for the following operations **Books**

ID Title price Status

1 The witcher 100 Available

2 Harry potter 200 Available

3 Nineteen eighty four 200 available

1 The witcher 100 available

Movies

ID title price status

1 Iron man 100 Not available

2 Harry potter 200 available

a) Returns all distinct rows from 2 tables. (UNION)

b) Returns all rows from 2 tables. (UNION ALL)

c) Returns all distinct rows common to both tables (INTERSECT)

d) Returns all rows common to both tables (INTERSECT ALL)

e) Returns all distinct rows from ‘books’ table that is not in ‘movies’ table.(EXCEPT) f) Returns all rows from ‘books’ table that is not in ‘movies’ table. (EXCEPT ALL)

Theory:

SQL set operators are used to combine the results obtained from two or more queries into a single result. The queries which contain two or more subqueries are known as compounded queries. There are four major types of SQL operators, namely: Union , Intersect ,Except. UNION:

· It combines distinct results of two or more SELECT statements

· Syntax for UNION in SQL is as follows:

SELECT column_name FROM table1_name UNION SELECT column_name FROM table2_name;

· Example for UNION in SQL:

SELECT name FROM customers_dec UNION SELECT name FROM customers_jan;

UNION ALL:

· The UNION set operator is used to combine all the results obtained from two or more SELECT statements. Unlike the Union operator, it considers duplicate values and includes them in the final result.

· Syntax for UNION ALL in SQL is as follows:

SELECT column_name FROM table1_name UNION ALL SELECT column_name FROM table2_name;

· Example for UNION ALL in SQL:

SELECT name FROM customers_dec UNION ALL SELECT name FROM customers_jan;

INTERSECT:

· The intersect set operator used to combine all the results of two SELECT statements. But returns only those records that are common to both the SELECT statements. · Syntax for INTERSECT in SQL is as follows:

SELECT column_name FROM table1_name INTERSECT SELECT column_name FROM table2_name;

· Example for INTERSECT in SQL:

SELECT column_name FROM table1_name INERSECT SELECT column_name FROM table2_name;

INTERSECT ALL:

· The intersect all set operator used to combine all the results of two SELECT statements. But returns only those records that are common to both the SELECT statements. . Unlike the inersect operator, it considers duplicate values and includes them in the final result. · Syntax for INTERSECT ALL in SQL is as follows:

SELECT column_name FROM table1_name INTERSECT ALL SELECT column_name FROM table2_name

· Example for INTERSECT ALL in SQL:

SELECT column_name FROM table1_name INERSECT ALL SELECT column_name FROM table2_name;

EXCEPT:

· The EXCEPT set operator used to combine all the results of two or more SELECT statements. But returns only those records that are present exclusively in the first table

· Syntax for EXCEPT in SQL is as follows:

SELECT column_name FROM table1_name EXCEPT SELECT column_name FROM table2_name;

· Example for EXCEPT in SQL:

SELECT column_name FROM table1_name EXCEPT SELECT column_name FROM table2_name;

EXCEPT ALL:

· The EXCEPT ALL set operator used to combine all the results of two or more SELECT statements. But returns only those records that are present exclusively in the first table. . Unlike the except operator, it considers duplicate values and includes them in the final result.

· Syntax for EXCEPT ALL in SQL is as follows:

SELECT column_name FROM table1_name EXCEPT ALL SELECT column_name FROM table2_name;

· Example for EXCEPT ALL in SQL:

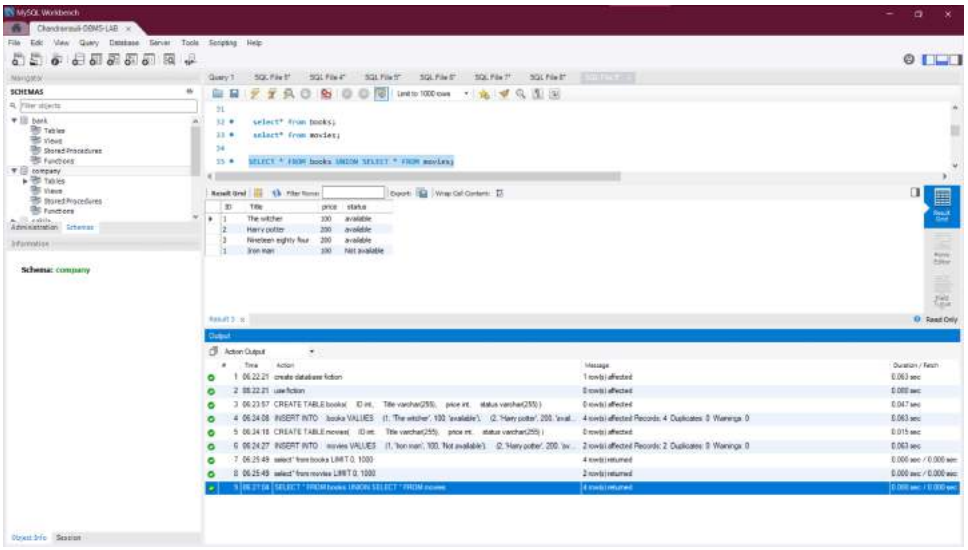
```
SELECT column_name FROM table1_name EXCEPT ALL SELECT column_name FROM table2_name;
```

Result: Set operations are familiarized and output is verified.

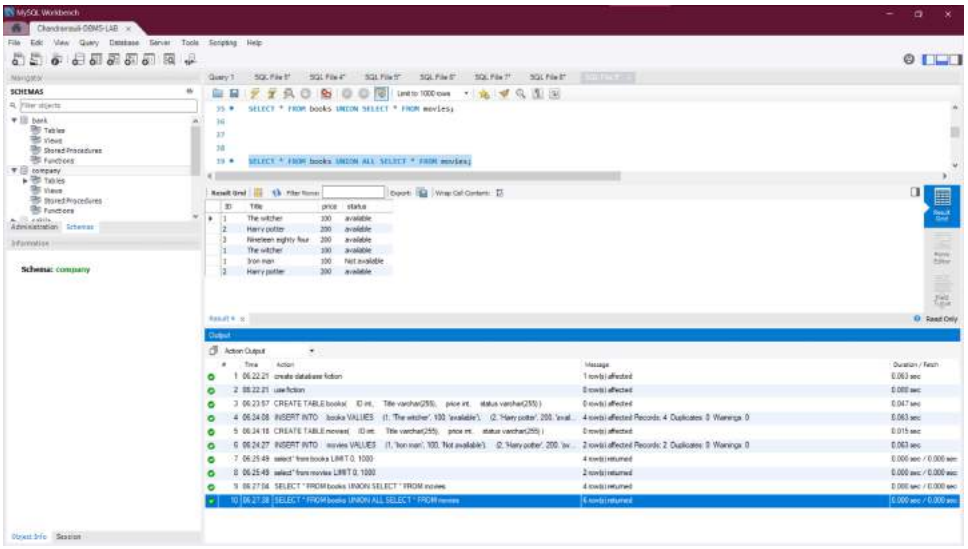
Remarks:(To be filled by faculty)

OUTPUT:

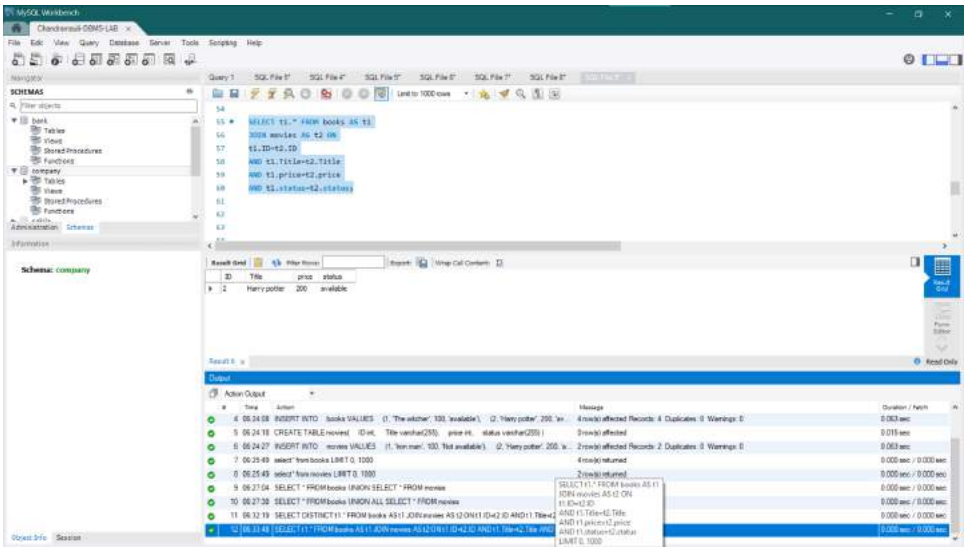
a) Returns all distinct rows from 2 tables. (UNION)



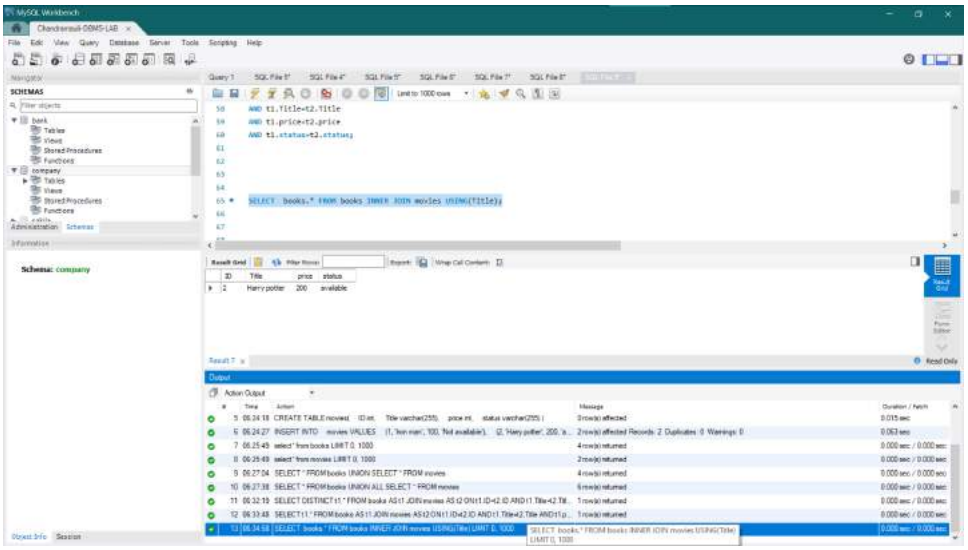
b) Returns all rows from 2 tables. (UNION ALL)



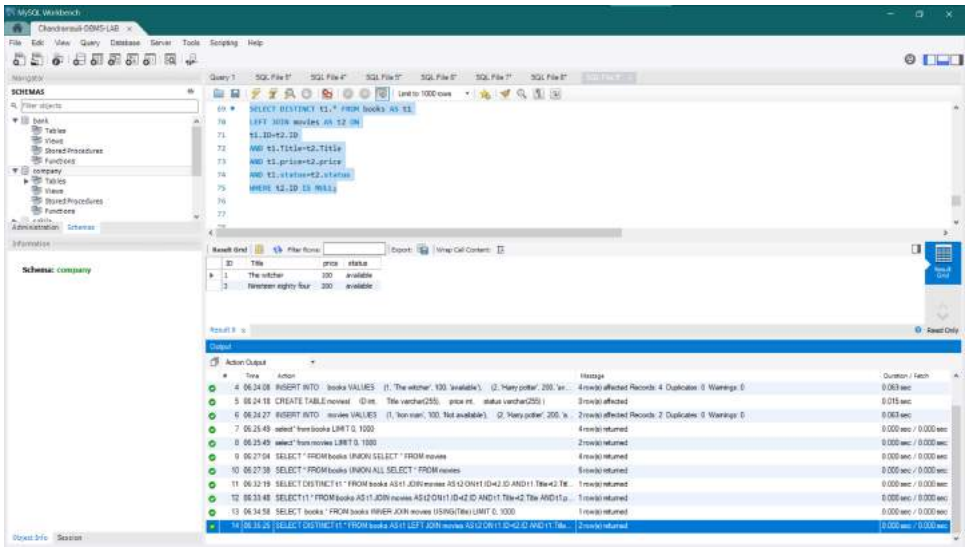
c) Returns all distinct rows common to both tables (INTERSECT)



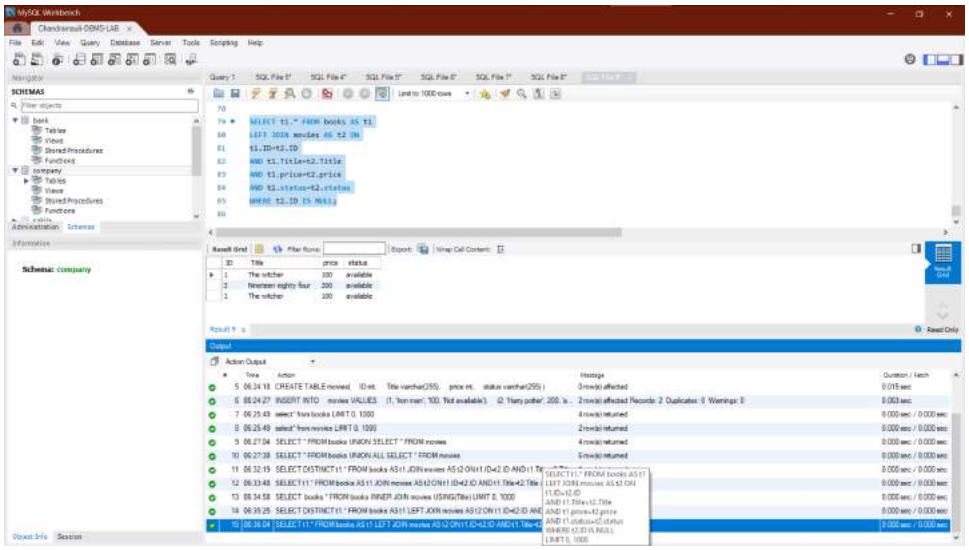
d) Returns all rows common to both tables (INTERSECT ALL)



e) Returns all distinct rows from ‘books’ table that is not in ‘movies’ table.(EXCEPT)



f) Returns all rows from ‘books’ table that is not in ‘movies’ table. (EXCEPT ALL)



(Left side of a page)