RATHINAM COLLEGE OF ARTS AND SCIENCE

(An Autonomous Institution Affiliated to Bharathiar University,
Accredited by NAAC with A++ (3.60 CGPA) in 3rd cycle,
NIRF Ranked, Approved by AICTE and recognized by UGC under 2(f) & 12B)
Rathinam Techzone Campus, Pollachi Road, Eachanari (PO), Coimbatore - 641021.
www.rathinamcollege.ac.in | info@rathinam.in | 0422-4040906

DEPARTMENT OF COMPUTER SCIENCE

RECORD NOTE BOOK

23BCS2DP – DISCIPLINE SPECIFIC CORE PRACTICAL-RDBMS

NAME	:
REGISTER NUMBER	:
YEAR/SEMESTER	:
ACADEMIC YEAR	:

Celebrate life

RATHINAM

RATHINAM COLLEGE OF ARTS AND SCIENCE

(An Autonomous Institution Affiliated to Bharathiar University,
Accredited by NAAC with A++ (3.60 CGPA) in 3rd cycle,
NIRF Ranked, Approved by AICTE and recognized by UGC under 2(f) & 12B)
Rathinam Techzone Campus, Pollachi Road, Eachanari (PO), Coimbatore - 641021.
www.rathinamcollege.ac.in | info@rathinam.in | 0422-4040906

BONAFIDE CERTIFICATE

NAME	:
ACADEMIC YEAR	:
YEAR/SEMESTER	:
BRANCH	:

UNIVERSITY REGISTER NUMBER: ………………………………..

Certified that this is the bonafide record of work done by the above student in the

_____________________________________Laboratory during the year 2023-2024.

Head of the Department **Staff-in-Charge**

Submitted for the Practical Examination held on __________________________

Internal Examiner **External Examiner**

S.No	Date	Experiment Name	Marks	Staff Sign

S.No	Date	Experiment Name	Marks	Staff Sign

EX NO: 1	**DDL, DML, DQL, TCL**
DATE:	

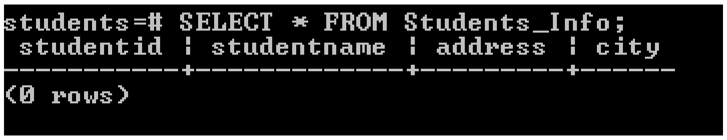
Aim:

To Working with SQL commands –Data Definition Commands, Data Manipulation Commands for inserting, deleting, updating and retrieving Tables and Transaction Control statements

Procedure:

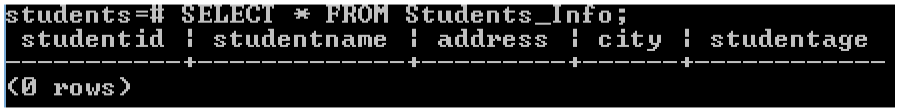
1.1 Data Definition Language (DDL)

```
Postgres=#  CREATE TABLE Students_Info
   (
   StudentID int,
   StudentName varchar(255),
   Address varchar(255),
   City varchar(255)
   );
CREATE TABLE
Postgres=#  SELECT * FROM Students_Info;
```



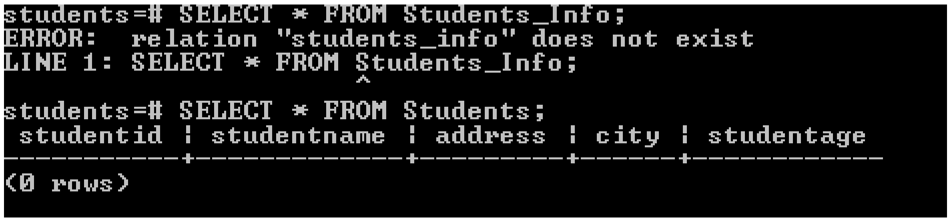
```
students=# SELECT * FROM Students_Info;
 studentid | studentname | address | city
-----------+-------------+---------+------
(0 rows)
```

```
Postgres=#  ALTER TABLE students_info ADD studentAge int;
ALTER TABLE
```



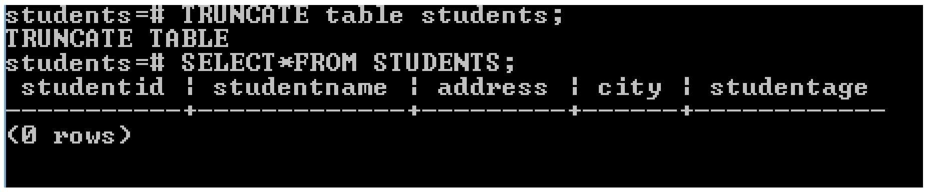
```
students=# SELECT * FROM Students_Info;
 studentid | studentname | address | city | studentage
-----------+-------------+---------+------+------------
(0 rows)
```

```
Postgres=#  ALTER TABLE students_info RENAME to Students;
ALTER TABLE
```



```
students=# SELECT * FROM Students_Info;
ERROR:  relation "students_info" does not exist
LINE 1: SELECT * FROM Students_Info;
                      ^
students=# SELECT * FROM Students;
 studentid | studentname | address | city | studentage
-----------+-------------+---------+------+------------
(0 rows)
```

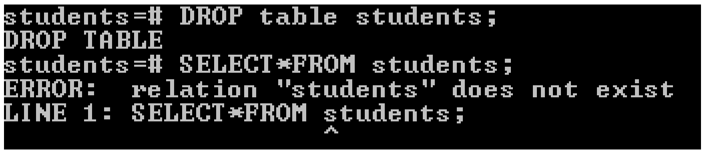
```
Postgres=#  TRUNCATE table students_info;
TRUNCATE TABLE
```



```
students=# TRUNCATE table students;
TRUNCATE TABLE
students=# SELECT*FROM STUDENTS;
 studentid | studentname | address | city | studentage
-----------+-------------+---------+------+------------
(0 rows)
```

Postgres=# DROP table students_info;

DROP TABLE



```
students=# DROP table students;
DROP TABLE
students=# SELECT*FROM students;
ERROR:  relation "students" does not exist
LINE 1: SELECT*FROM students;
                    ^
```

1.2 Data Manipulation Language (DML)

CREATE TABLE   Students_Info

(

StudentID int,

StudentName varchar(255),

Address varchar(255),

City varchar(255)

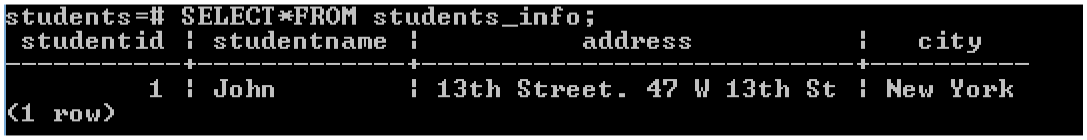
);

CREATE TABLE

INSERT INTO students_info(studentid, studentname ,address,city)

VALUES (01,'John','13th Street. 47 W 13th St' , 'New York');

INSERT 0 1



```
students=# SELECT*FROM students_info;
 studentid | studentname |          address          |   city
-----------+-------------+---------------------------+----------
         1 | John        | 13th Street. 47 W 13th St | New York
(1 row)
```

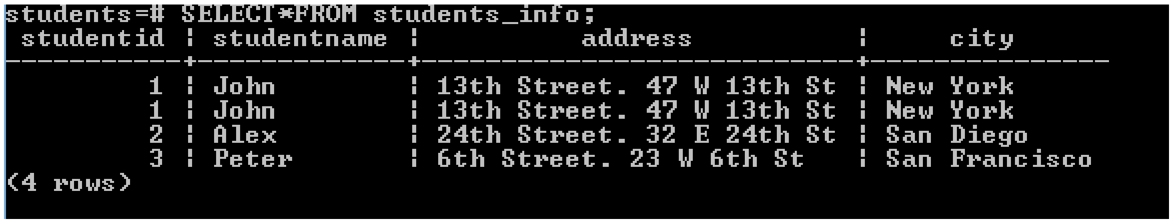
INSERT INTO students_info(studentid, studentname ,address,city)

VALUES (01,'John','13th Street. 47 W 13th St' , 'New York'),

(02,'Alex','24th Street. 32 E 24th St' , 'San Diego'),

(03,'Peter','6th Street. 23 W 6th St' , 'San Francisco');

INSERT 0 3

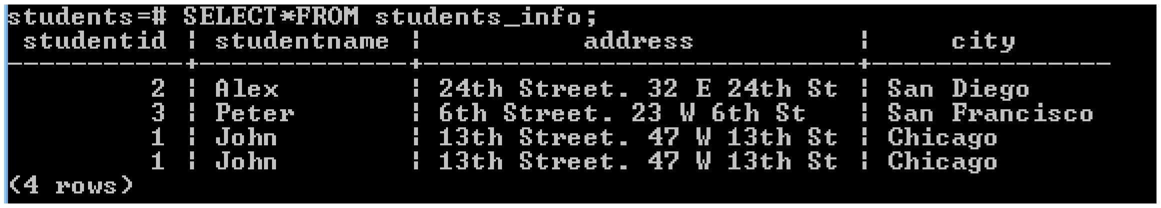

```
students=# SELECT*FROM students_info;
 studentid | studentname |          address           |     city
-----------+-------------+----------------------------+---------------
         1 | John        | 13th Street. 47 W 13th St  | New York
         1 | John        | 13th Street. 47 W 13th St  | New York
         2 | Alex        | 24th Street. 32 E 24th St  | San Diego
         3 | Peter       | 6th Street. 23 W 6th St    | San Francisco
(4 rows)
```

UPDATE students_info

SET city = 'Chicago'

WHERE studentid = 1;

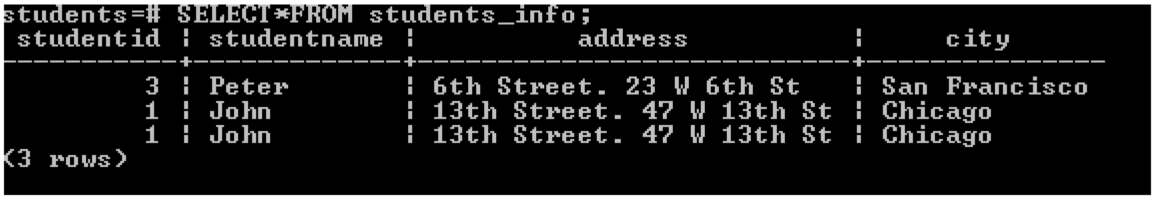
UPDATE 2


```
students=# SELECT*FROM students_info;
 studentid | studentname |          address           |     city
-----------+-------------+----------------------------+---------------
         2 | Alex        | 24th Street. 32 E 24th St  | San Diego
         3 | Peter       | 6th Street. 23 W 6th St    | San Francisco
         1 | John        | 13th Street. 47 W 13th St  | Chicago
         1 | John        | 13th Street. 47 W 13th St  | Chicago
(4 rows)
```

DELETE FROM students_info

WHERE studentid = 2;

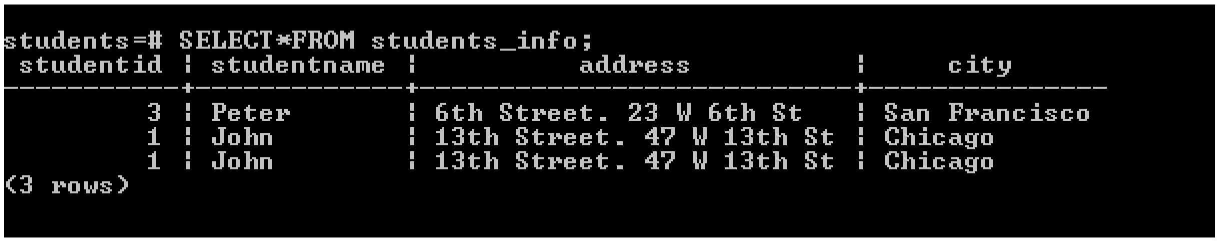
DELETE 1


```
students=# SELECT*FROM students_info;
 studentid | studentname |          address           |     city
-----------+-------------+----------------------------+---------------
         3 | Peter       | 6th Street. 23 W 6th St    | San Francisco
         1 | John        | 13th Street. 47 W 13th St  | Chicago
         1 | John        | 13th Street. 47 W 13th St  | Chicago
(3 rows)
```

1.3 Data Query Language (DQL)

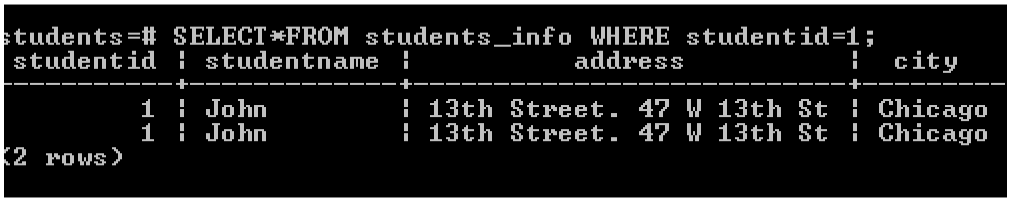
DISPLAY ALL INFORMATION FROM STUDENTS INFORMATION TABLE.

select * from students_info;


```
students=# SELECT*FROM students_info;
 studentid | studentname |          address           |     city
-----------+-------------+----------------------------+---------------
         3 | Peter       | 6th Street. 23 W 6th St    | San Francisco
         1 | John        | 13th Street. 47 W 13th St  | Chicago
         1 | John        | 13th Street. 47 W 13th St  | Chicago
(3 rows)
```

DISPLAY ALL INFORMATION FROM STUDENTS TABLE WHEN STUDENT ID IS 1

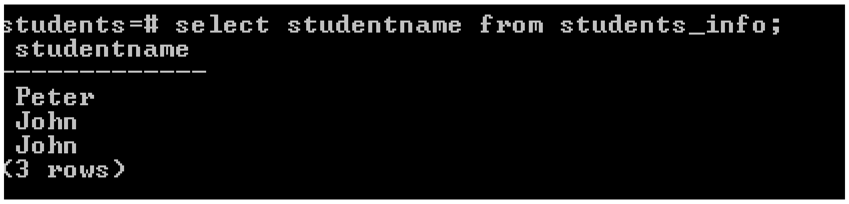
select * from students_info where studentid=1;



```
tudents=# SELECT*FROM students_info WHERE studentid=1;
 studentid | studentname |          address           |  city
-----------+-------------+----------------------------+---------
         1 | John        | 13th Street. 47 W 13th St | Chicago
         1 | John        | 13th Street. 47 W 13th St | Chicago
(2 rows)
```

DISPLAY STUDENTS NAMES FROM STUDENT TABLE

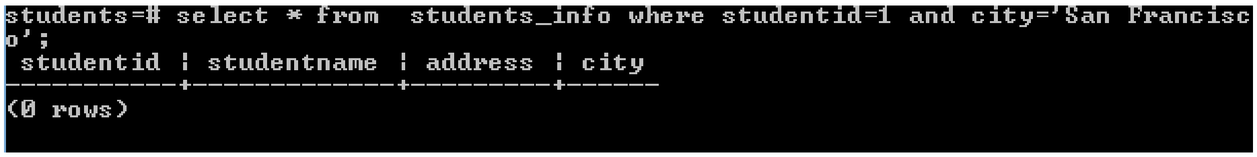
select studentname from students_info;



```
students=# select studentname from students_info;
 studentname
-------------
 Peter
 John
 John
(3 rows)
```

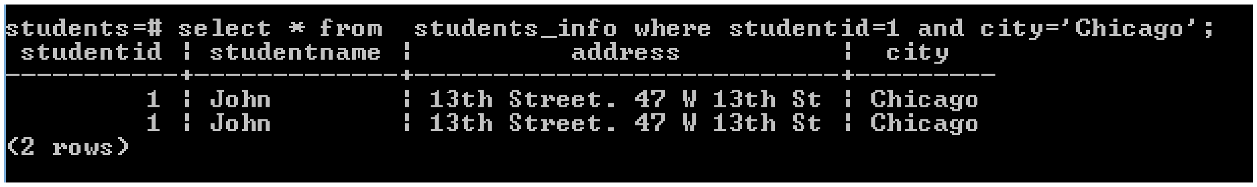
DISPLAY ALL INFORMATION FROM STUDENTS TABLE WHEN STUDENT ID IS 1 AND CITY IS 'SAN FRANCISCO'

select * from students_info where studentid=1 and city='San Francisco';



```
students=# select * from  students_info where studentid=1 and city='San Francisc
o';
 studentid | studentname | address | city
-----------+-------------+---------+------
(0 rows)
```

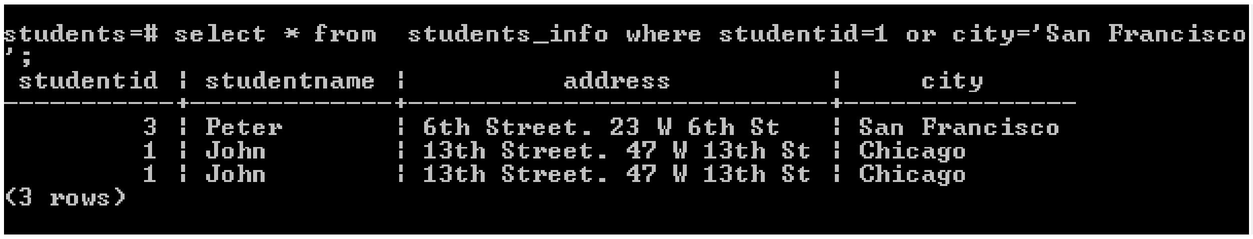
select * from students_info where studentid=1 and city='Chicago';



```
students=# select * from  students_info where studentid=1 and city='Chicago';
 studentid | studentname |          address           |  city
-----------+-------------+----------------------------+---------
         1 | John        | 13th Street. 47 W 13th St | Chicago
         1 | John        | 13th Street. 47 W 13th St | Chicago
(2 rows)
```

DISPLAY ALL INFORMATION FROM STUDENTS TABLE WHEN STUDENT ID IS 1 OR CITY IS SAN FRANCISCO

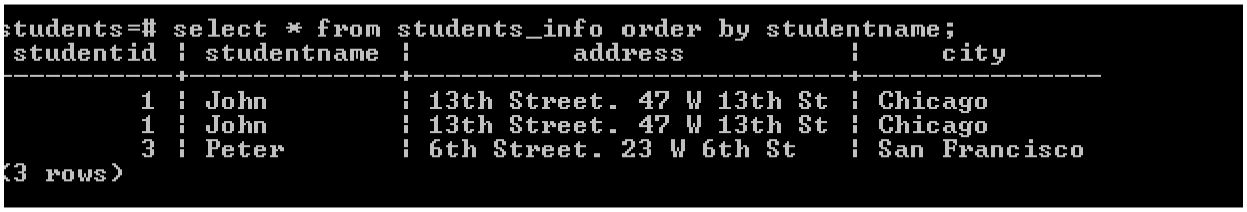
select * from students_info where studentid=1 or city='San Francisco';



```
students=# select * from  students_info where studentid=1 or city='San Francisco
';
 studentid | studentname |          address           |     city
-----------+-------------+----------------------------+---------------
         3 | Peter       | 6th Street. 23 W 6th St    | San Francisco
         1 | John        | 13th Street. 47 W 13th St  | Chicago
         1 | John        | 13th Street. 47 W 13th St  | Chicago
(3 rows)
```

DISPLAY STUDENT NAMES IN ASCENDING ORDER

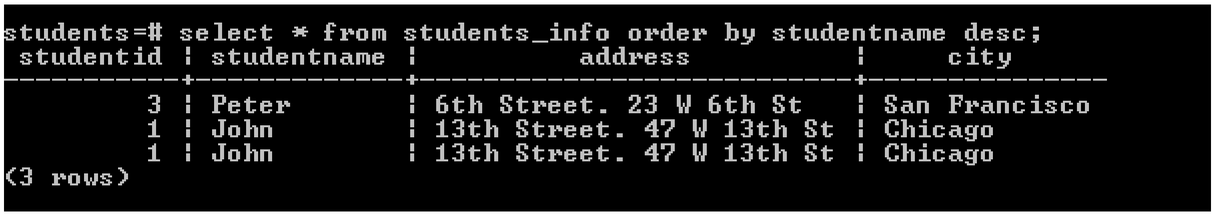
select * from students_info order by studentname;



```
students=# select * from students_info order by studentname;
 studentid | studentname |          address           |     city
-----------+-------------+----------------------------+---------------
         1 | John        | 13th Street. 47 W 13th St  | Chicago
         1 | John        | 13th Street. 47 W 13th St  | Chicago
         3 | Peter       | 6th Street. 23 W 6th St    | San Francisco
(3 rows)
```

DISPLAY STUDENT NAMES IN DESCENDING ORDER

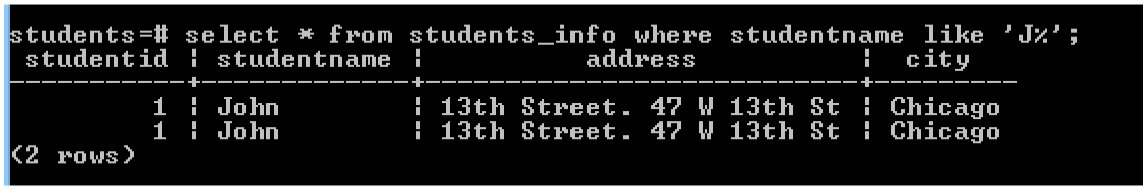
select * from students_info order by studentname desc;



```
students=# select * from students_info order by studentname desc;
 studentid | studentname |          address           |     city
-----------+-------------+----------------------------+---------------
         3 | Peter       | 6th Street. 23 W 6th St    | San Francisco
         1 | John        | 13th Street. 47 W 13th St  | Chicago
         1 | John        | 13th Street. 47 W 13th St  | Chicago
(3 rows)
```

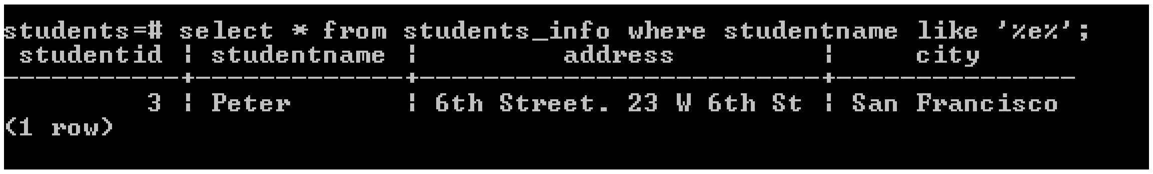
DISPLAY STUDENT NAMES WHO HAVE NAMES STARTING WITH LETTER J

select * from students_info where studentname like 'J%';



```
students=# select * from students_info where studentname like 'J%';
 studentid | studentname |          address           |  city
-----------+-------------+----------------------------+---------
         1 | John        | 13th Street. 47 W 13th St  | Chicago
         1 | John        | 13th Street. 47 W 13th St  | Chicago
(2 rows)
```

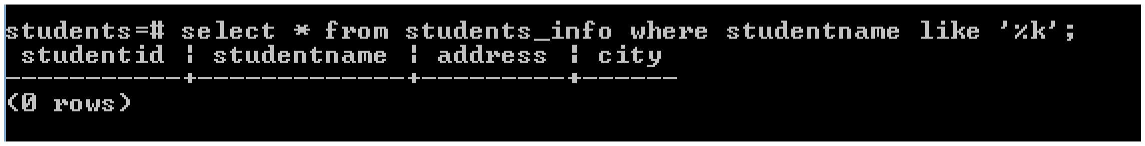
select * from students_info where studentname like '%e%';



```
students=# select * from students_info where studentname like '%e%';
 studentid | studentname |          address         |     city
-----------+-------------+--------------------------+---------------
         3 | Peter       | 6th Street. 23 W 6th St  | San Francisco
(1 row)
```

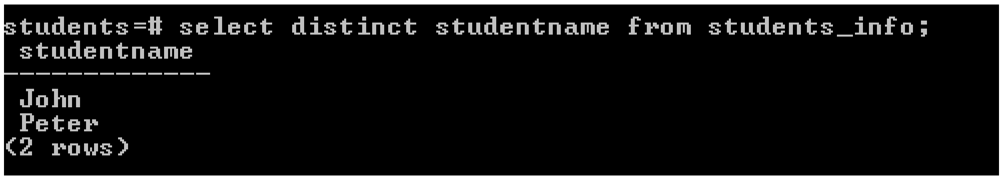
DISPLAY STUDENTS NAMES WHO HAVE NAMES ENDING WITH LETTER 'K'

select * from students_info where studentname like '%k';


```
students=# select * from students_info where studentname like '%k';
 studentid | studentname | address | city
-----------+-------------+---------+------
(0 rows)
```

DISPLAY STUDENTNAME FROM STUDENT TABLE WITHOUT DUPLICATION

select distinct studentname from students_info;


```
students=# select distinct studentname from students_info;
 studentname
-------------
 John
 Peter
(2 rows)
```

1.4 Transaction Control Language (TCL)

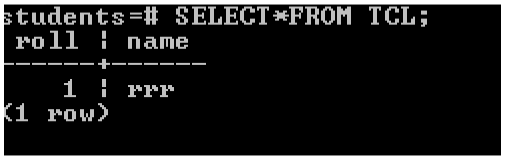
create table tcl(roll int, name varchar(35));

CREATE TABLE

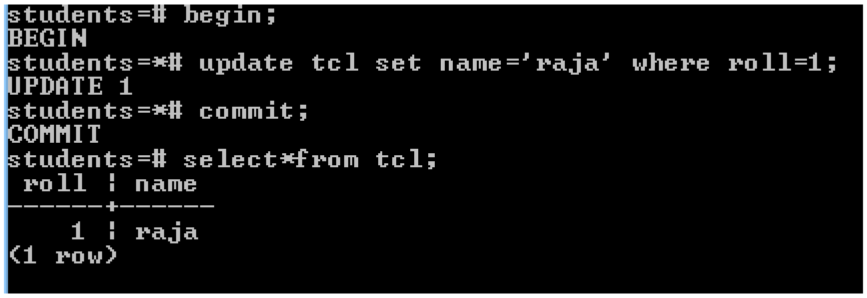
insert into tcl values(1, 'rrr');

INSERT 0 1

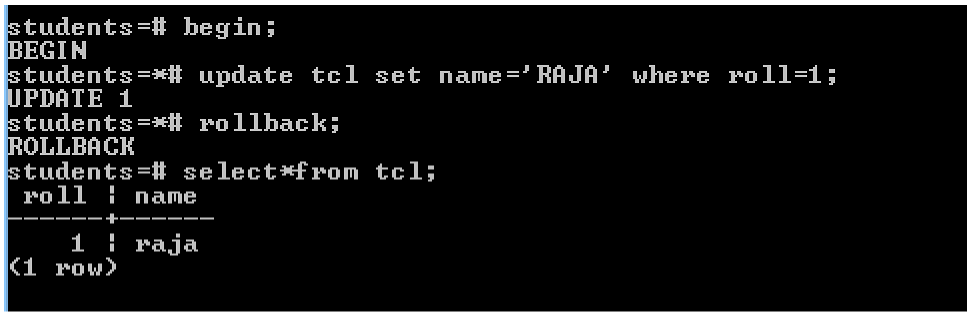
select * from tcl;


```
students=# SELECT*FROM TCL;
 roll | name
------+------
    1 | rrr
(1 row)
```

Commit:


```
students=# begin;
BEGIN
students=*# update tcl set name='raja' where roll=1;
UPDATE 1
students=*# commit;
COMMIT
students=# select*from tcl;
 roll | name
------+------
    1 | raja
(1 row)
```

Rollback:



```
students=# begin;
BEGIN
students=*# update tcl set name='RAJA' where roll=1;
UPDATE 1
students=*# rollback;
ROLLBACK
students=# select*from tcl;
 roll | name
------+------
    1 | raja
(1 row)
```

Result:

Thus the all DDL, DML, DQL, TCL commands are executed and verified successfully.

Ex.No: 2	**Simple Queries, Nested Queries, Sub Queries and Joins**

Aim:

To execute and verify the SQL commands using Simple Queries, Nested Queries, Sub Queries and Joins.

Procedure:

Simple and sub queries:

create table customers (id int , name varchar(20) , age int , address varchar(20) , salary numeric(5)) ;

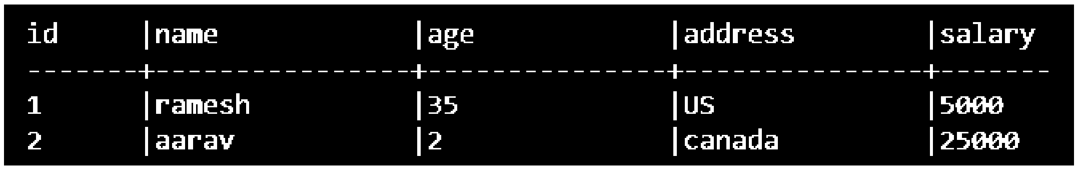
CREATE TABLE

insert into customers values(1, 'ramesh', 35,'erode', 5000) ;

insert into customers values(2, 'aarav', 2,'canada', 25000) ;

INSERT 0 2

select *from customers where id in (select id from customers where salary > 10000);

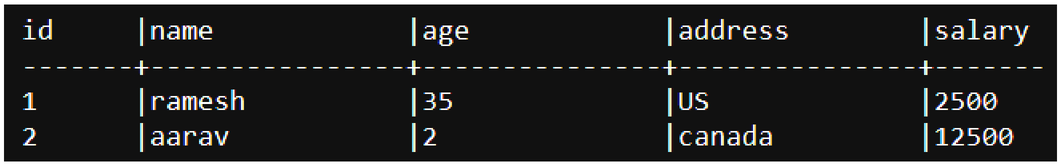


```
id      |name             |age              |address          |salary
-------+-----------------+-----------------+-----------------+-------
1       |ramesh           |35               |US               |5000
2       |aarav            |2                |canada           |25000
```

update customers set salary = salary * 0.50 where age in (select age from customers where age = 2);

UPDATE

select * from customers



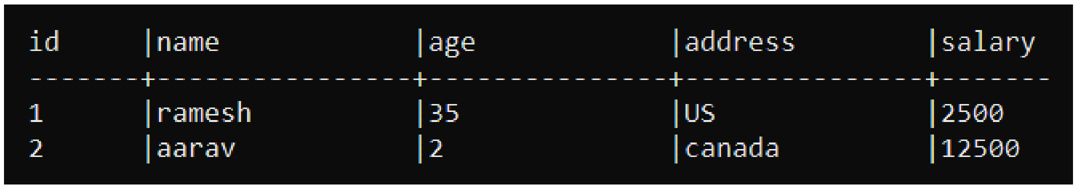
```
id      |name             |age              |address          |salary
-------+-----------------+-----------------+-----------------+-------
1       |ramesh           |35               |US               |2500
2       |aarav            |2                |canada           |12500
```

create table customers_bkp (id int , name varchar(20) , age int , address v archar(20) , salary numeric(5)) ;

CREATE TABLE

insert into customers_bkp select * from customers where id in (select id from customers) ;

select * from customers_bkp

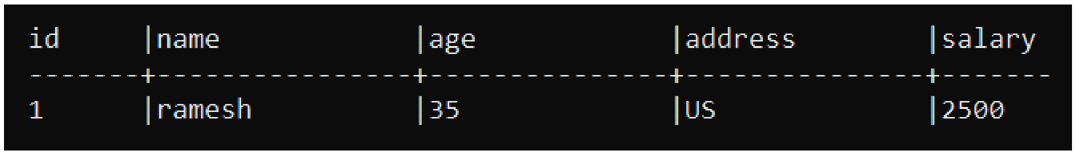


```
id      |name               |age                |address            |salary
--------+-------------------+-------------------+-------------------+--------
1       |ramesh             |35                 |US                 |2500
2       |aarav              |2                  |canada             |12500
```

delete from customers where age in (select age from customers_bkp where age = 35);

DELETE 1

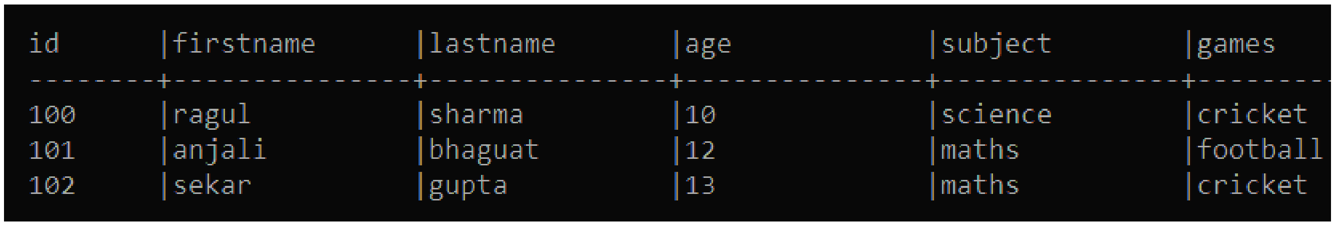
select * from customers_bkp;



```
id      |name               |age                |address            |salary
--------+-------------------+-------------------+-------------------+--------
1       |ramesh             |35                 |US                 |2500
```

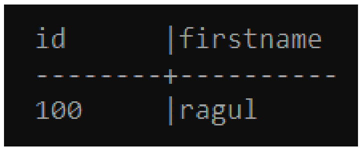
<u>Nested Queries:</u>

select * from studentdetail;



```
id      |firstname        |lastname         |age              |subject          |games
--------+-----------------+-----------------+-----------------+-----------------+---------
100     |ragul            |sharma           |10               |science          |cricket
101     |anjali           |bhaguat          |12               |maths            |football
102     |sekar            |gupta            |13               |maths            |cricket
```

select id,firstname from studentdetail where firstname in(select • firstname from studentdetail where subject='science');



```
id      |firstname
--------+----------
100     |ragul
```

create table mathsgroup(id number(10),name varchar(15));

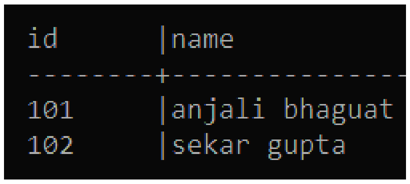
CREATE TABLE

TO GROUP ALL THE STUDENTS WHO STUDY MATHS IN A TABLE MATHSGROUP

insert into mathsgroup(id,name) select id,firstname || lastname from studentdetail where subject='maths';

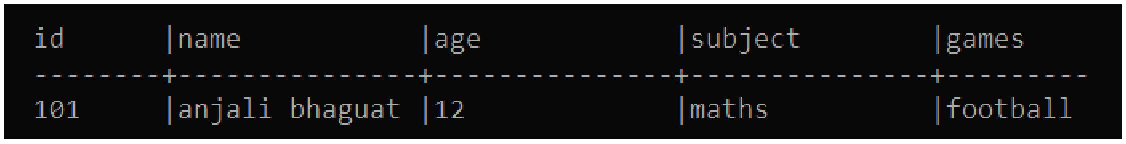
INSERT 0 1

select * from mathsgroup;



```
id      |name
--------+--------------
101     |anjali bhaguat
102     |sekar gupta
```

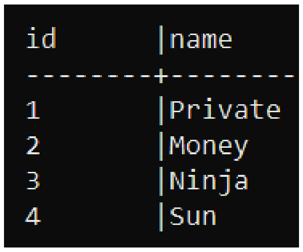
select id, (select name from mathsgroup where id=101) as name, age, subject, games from studentdetail where id=101;



```
id      |name           |age            |subject        |games
--------+---------------+---------------+---------------+---------
101     |anjali bhaguat |12             |maths          |football
```

<u>Joins:</u>

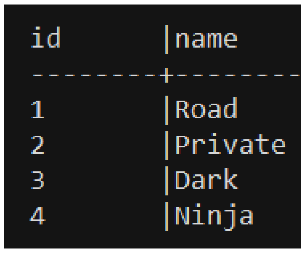
select *from table_a;



```
id      |name
--------+--------
1       |Private
2       |Money
3       |Ninja
4       |Sun
```

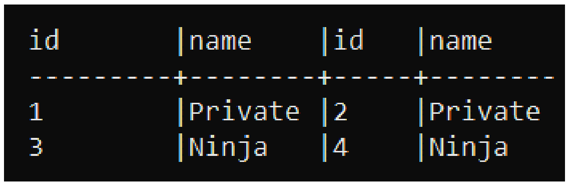
----

select *from table_b;



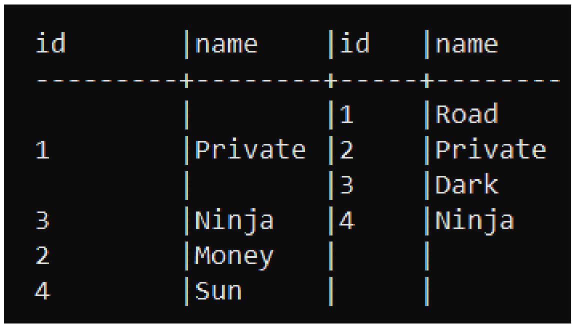
```
id      |name
--------+--------
1       |Road
2       |Private
3       |Dark
4       |Ninja
```

SELECT * FROM TABLE_A **INNER JOIN** TABLE_B ON TABLE_A.NAME=TABLE_B.NAME;



```
id       |name    |id   |name
---------+--------+-----+--------
1        |Private |2    |Private
3        |Ninja   |4    |Ninja
```

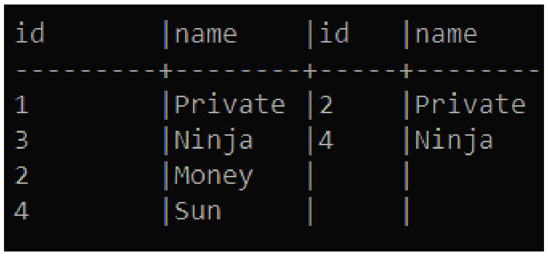
SELECT * FROM TABLE_A **FULL OUTER JOIN** TABLE_B ON TABLE_A.NAME=TABLE_B.NAME;



```
id       |name    |id   |name
---------+--------+-----+--------
         |        |1    |Road
1        |Private |2    |Private
         |        |3    |Dark
3        |Ninja   |4    |Ninja
2        |Money   |     |
4        |Sun     |     |
```

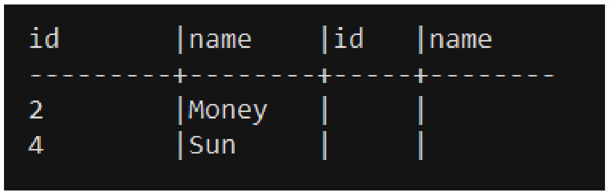
6 rows selected.

SELECT * FROM TABLE_A **LEFT OUTER JOIN** TABLE_B ON TABLE_A.NAME=TABLE_B.NAME;



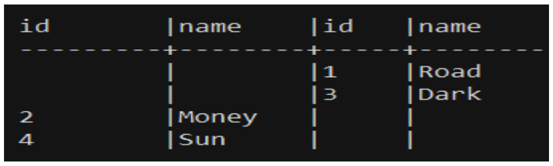
```
id       |name    |id   |name
---------+--------+-----+--------
1        |Private |2    |Private
3        |Ninja   |4    |Ninja
2        |Money   |     |
4        |Sun     |     |
```

SELECT * FROM TABLE_A **LEFT OUTER JOIN** TABLE_B ON TABLE_A.NAME=TABLE_B.NAME WHERE TABLE_B.ID IS **NULL**;



```
id       |name    |id   |name
---------+--------+-----+--------
2        |Money   |     |
4        |Sun     |     |
```

SELECT * FROM TABLE_A **FULL OUTER JOIN** TABLE_B ON TABLE_A.NAME=TABLE_B.NAME WHERE TABLE_A.ID IS **NULL OR** TABLE_B.ID IS **NULL**;



```
id       |name    |id   |name
---------+--------+-----+--------
         |        |1    |Road
         |        |3    |Dark
2        |Money   |     |
4        |Sun     |     |
```

<u>**Result:**</u>

Thus the SQL commands using Simple Queries, Nested Queries, Sub Queries and Joins are executed and verified successfully.

Ex.No: 3	**Views and Sequences**

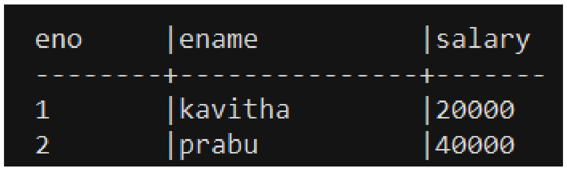
Aim:

To execute and verify the SQL commands for Views and Sequences.

Procedure:

Views:

select * from employee;



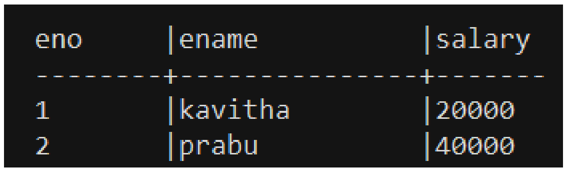
```
eno     |ename          |salary
--------+---------------+-------
1       |kavitha        |20000
2       |prabu          |40000
```

CREATING VIEWS AND DML OPERATIONS WITH VIEWS.

create view employeeview as select * from employee;

CREATE VIEW

select * from employeeview;

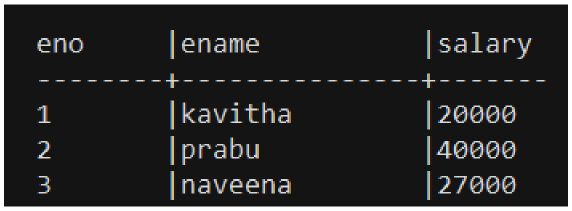


```
eno     |ename          |salary
--------+---------------+-------
1       |kavitha        |20000
2       |prabu          |40000
```

insert into employeeview values(3,'naveena',27000)

INSERT 0 1

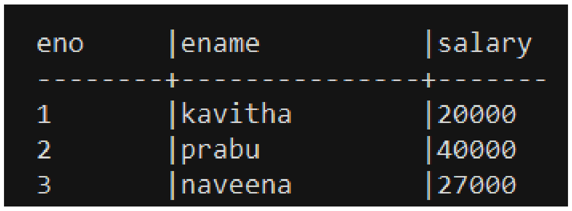
select * from employeeview;



```
eno     |ename          |salary
--------+---------------+-------
1       |kavitha        |20000
2       |prabu          |40000
3       |naveena        |27000
```

3 rows selected

select * from employee;



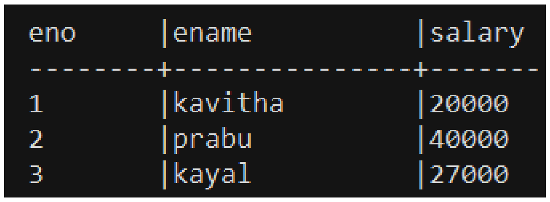
```
eno     |ename           |salary
--------+----------------+-------
1       |kavitha         |20000
2       |prabu           |40000
3       |naveena         |27000
```

3 rows selected.

update employeeview set ename='kayal' where eno=3;

UPDATE 1

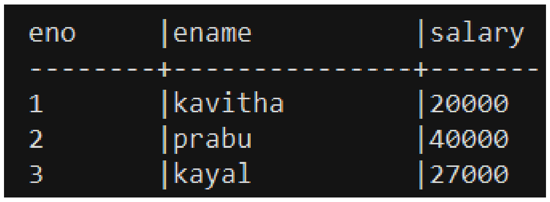
select * from employeeview;



```
eno     |ename           |salary
--------+----------------+-------
1       |kavitha         |20000
2       |prabu           |40000
3       |kayal           |27000
```

3 rows selected.

select * from employee;



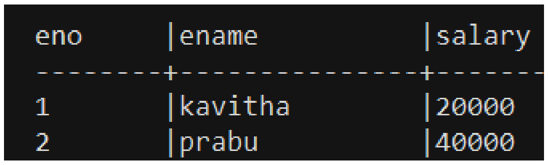
```
eno     |ename           |salary
--------+----------------+-------
1       |kavitha         |20000
2       |prabu           |40000
3       |kayal           |27000
```

3 rows selected.

delete from employeeview where eno=3;

DELETE 1

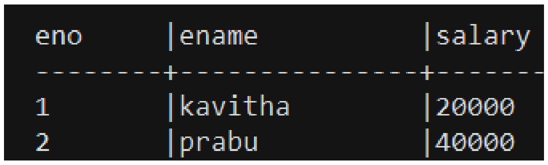
select * from employeeview;



```
eno     |ename          |salary
--------+---------------+-------
1       |kavitha        |20000
2       |prabu          |40000
```

2 rows selected.

select * from employee;



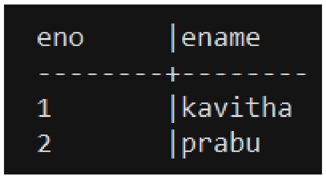
```
eno     |ename          |salary
--------+---------------+-------
1       |kavitha        |20000
2       |prabu          |40000
```

2 rows selected.

create view empview as select eno,ename from employee with **read only**;

CREATE VIEW

select * from empview;



```
eno     |ename
--------+--------
1       |kavitha
2       |prabu
```

2 rows selected.

insert into empview values(7,'suriya') ERROR at line 1:

ORA-01733: virtual column not allowed here

SEQUENCE:

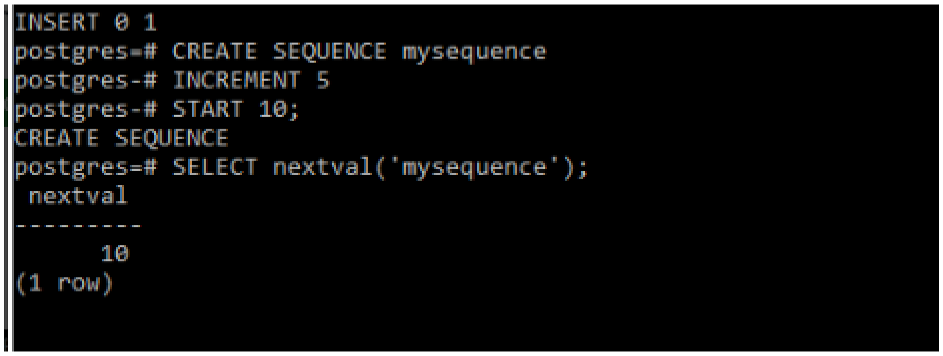
CREATE SEQUENCE mysequence

INCREMENT 5

START 10;

CREATE SEQUENCE

SELECT nextval('mysequence');



```
INSERT 0 1
postgres=# CREATE SEQUENCE mysequence
postgres-# INCREMENT 5
postgres-# START 10;
CREATE SEQUENCE
postgres=# SELECT nextval('mysequence');
 nextval
---------
      10
(1 row)
```

CREATE SEQUENCE three

INCREMENT -1

MINVALUE 1

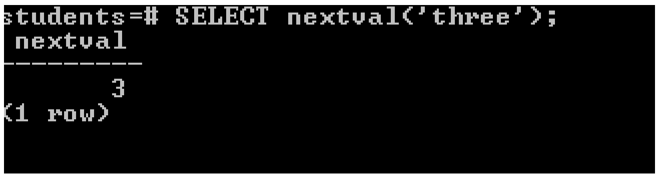
MAXVALUE 3

START 3

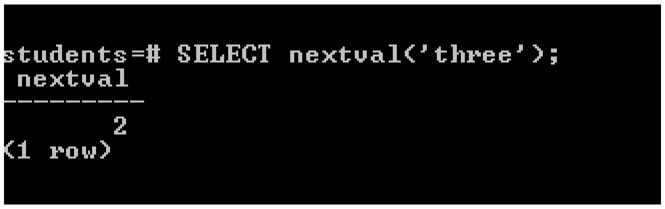
CYCLE;

CREATE SEQUENCE

SELECT nextval('three');



```
students=# SELECT nextval('three');
 nextval
---------
       3
(1 row)
```



```
students=# SELECT nextval('three');
 nextval
---------
       2
(1 row)
```

CREATE TABLE order_details(

order_id SERIAL,

item_id INT NOT NULL,

product_id INT,

product_name TEXT NOT NULL,

price DEC(10, 2) NOT NULL,

PRIMARY KEY(order_id, item_id)

);

CREATE TABLE

CREATE SEQUENCE order_item_id

START 10

INCREMENT 10

MINVALUE 10

OWNED BY order_details.item_id;

CREATE SEQUENCE

INSERT INTO

order_details(order_id, item_id, product_name, price)

VALUES

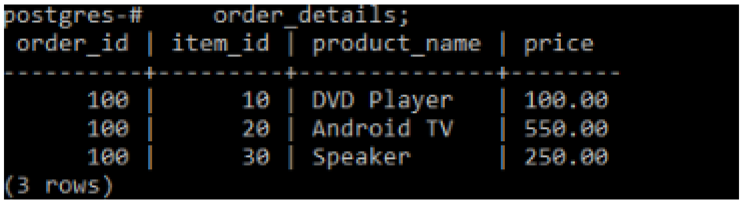
(100, nextval('order_item_id'), 'DVD Player', 100),

(100, nextval('order_item_id'), 'Android TV', 550),

(100, nextval('order_item_id'), 'Speaker', 250);

INSERT 3 0

```
SELECT
    order_id,
    item_id,
    product_name,
    price
FROM
    order_details;
```



```
postgres-#     order_details;
 order_id | item_id | product_name | price
----------+---------+--------------+--------
      100 |      10 | DVD Player   | 100.00
      100 |      20 | Android TV   | 550.00
      100 |      30 | Speaker      | 250.00
(3 rows)
```

RESULT:

Thus execute and verify the SQL commands for Views, Sequences.

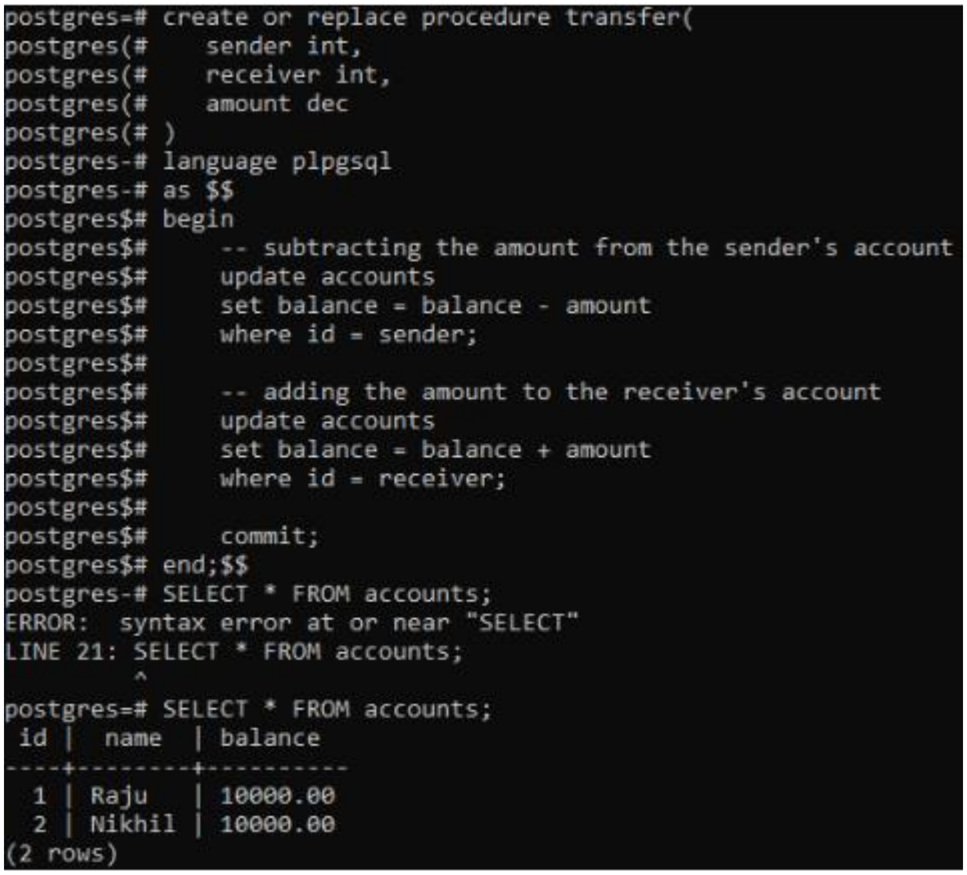
Ex.No: 4	**Stored Procedures and Functions**

Aim:

To execute and verify the SQL commands for Creation of Procedures and Functions.

Procedure:

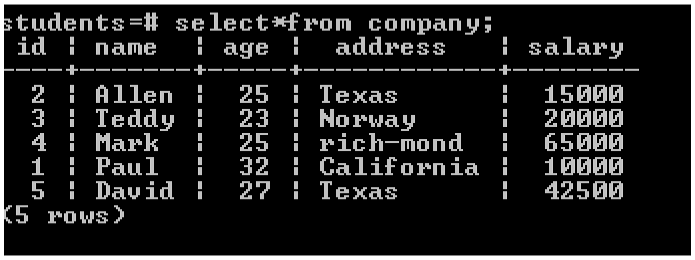
```
create table accounts (
    id int generated by default as identity,
    name varchar(100) not null,
    balance dec(15, 2) not null,
    primary key(id)
);
CREATE TABLE
insert into accounts(name, balance)
values('Raju', 10000);
insert into accounts(name, balance)
values('Nikhil', 10000);
INSERT 0 2

create or replace procedure transfer(
   sender int,
   receiver int,
   amount dec
)
language plpgsql
as $$
begin
    -- subtracting the amount from the sender's account
    update accounts
    set balance = balance - amount
    where id = sender;

    -- adding the amount to the receiver's account
    update accounts
    set balance = balance + amount
    where id = receiver;

    commit;
end;$$

call transfer(1, 2, 1000);

SELECT * FROM accounts;
```



```
postgres=# create or replace procedure transfer(
postgres(#    sender int,
postgres(#    receiver int,
postgres(#    amount dec
postgres(# )
postgres-# language plpgsql
postgres-# as $$
postgres$# begin
postgres$#     -- subtracting the amount from the sender's account
postgres$#     update accounts
postgres$#     set balance = balance - amount
postgres$#     where id = sender;
postgres$#
postgres$#     -- adding the amount to the receiver's account
postgres$#     update accounts
postgres$#     set balance = balance + amount
postgres$#     where id = receiver;
postgres$#
postgres$#     commit;
postgres$# end;$$
postgres-# SELECT * FROM accounts;
ERROR:  syntax error at or near "SELECT"
LINE 21: SELECT * FROM accounts;
         ^
postgres=# SELECT * FROM accounts;
 id |  name  | balance
----+--------+----------
  1 | Raju   | 10000.00
  2 | Nikhil | 10000.00
(2 rows)
```

Functions:

```
select*from company;
```



```
students=# select*from company;
 id | name  | age |  address    | salary
----+-------+-----+-------------+--------
  2 | Allen |  25 | Texas       |  15000
  3 | Teddy |  23 | Norway      |  20000
  4 | Mark  |  25 | rich-mond   |  65000
  1 | Paul  |  32 | California  |  10000
  5 | David |  27 | Texas       |  42500
(5 rows)
```

CREATE OR REPLACE FUNCTION totalRecords ()

RETURNS integer AS $total$

declare

total integer;

BEGIN

SELECT count(*) into total FROM COMPANY;

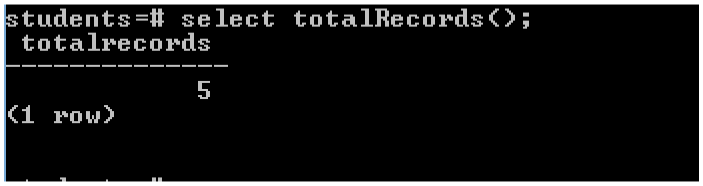
RETURN total;

END;

$total$ LANGUAGE plpgsql;

CREATE FUNCTION

select totalRecords();



```
students=# select totalRecords();
 totalrecords
--------------
            5
(1 row)
```

<u>RESULT:</u>

Thus execute and verify the SQL commands for Creation of functions and Procedures.