Program 1:

Create a table customer (cust_novarchar (5), cust_namevarchar (15), age number,phone varchar (10)).

```
create table customer(cust_novarchar(5),cust_name varchar(15),age int,phone varchar(10));
```

a) insert 5 records and display it.

```
insert into customer values(100,'amal',18,'1234567890');
insert into customer values(101,'kavya',19,'2345678901');
insert into customer values(102,'jwala',20,'3456789012');
insert into customer values(103,'abay',20,'4567890123');
insert into customer values(104,'abhi',19,'5678901234');

select * from customer;
```

b) add new field d_birth with date datatype.

```
alter table customer add d_birth date;
```

c) create another table cust_phone with fields cust_name and phone from customertable.

create table cust_phone as(select cust_name,phone from customer);

d) remove the field age.

alter table customer drop column age;

e) change the size of the cust_name to 25.

alter table customer alter column cust_name type varchar(25);

f) delete all the records from the table.

delete from customer;

g) rename the table cutomer to cust.

alter table customer rename to cust;

h) drop the table.

drop table cust;

Program 2:

Create a table sale_man(salesman_no primary key, s_name not null, place, phone unique).

```
create table sales_man(salesman_novarchar(5) primary key,s_name varchar(25) not null,place varchar(30),phone varchar(10) unique);
```

Create table sales_order (order_no primary key order_date not null salesman_no foreign key references salesman_no in sales_mandel_type values should be either P or F (check constraints) order_status values should be 'Inprocess','Fullfilled','Backorder', 'Cancelled' (check constraints)).

```
create table sales_order(order_no varchar(5) primary key,order_date date not null,salesman_no varchar(5),del_type char check(del_type in('P','F')),order_status varchar(15) check(order_status in('Inprocess','Fulfilled','Backorder','Cancelled')),foreign key(salesman_no) references sales_man(salesman_no));
```

a) Insert few records in both tables.

```
insert into sales_man values(100,'abay','thamarassery','1234567890');
insert into sales_man values(101,'riya','mukkam','2345678901');
insert into sales_man values(102,'aswin','areekode','3456789012');
insert into sales_man values(103,'harshal','kozhikode','4567890123');
insert into sales_order values(200,'08-02-2021',100,'P','Fulfilled');
insert into sales_order values(201,'07-02-2021',101,'F','Cancelled');
insert into sales_order values(202,'08-12-2021',102,'F','Backorder');
```

b) Delete primary key from sales_man table.

```
alter table sales_man drop constraint sales_manpkey cascade;
```

c) Delete Foreign key and Check constraints from sales_order table.

```
alter table saes_order drop constraint fk_salesman;
alter table sales_order drop constraint check_order_status;
alter table sales_order drop constraint check_del_type;
```

d) Add primary key in sales_man using ALTER TABLE.

```
alter table sales_man add constraint pk_sales_man primary key
s_name;
```

e) Add foreign key and CHECK constraints in sales_order table using ALTER TABLE.

```
alter table sales_order add constraint fk_salesman foreign
key(salesman_no) references sales_man(salesman_no);

alter table sales_order add constraint check_del_type(check(del_type
in (‘P’,’F’));
```

Program 3:

Create a table Hospital with the field(doctorid, doctorname, department, qualification, experience).

create table hospital(doctored varchar(15) primary key,doctorname varchar(30),department varchar(25),qualification varchar(30),experience int);

Write the queries to perform the following.

a) Insert 5 records.

insert into hospital values('103'.'riya','ENT','MS',2);

insert into hospital values('105'.'sachind','cardiac','MD',3);

insert into hospital values('107'.'maxwell','ortho','MD',1);

insert into hospital values('102'.'sharun','onco','MS',6);

insert into hospital values('104'.'harshal','dental','BDS',2);

b) Display the details of Doctors.

select doctorname,deparment from hospital;

c) Display the details of doctors who have the qualification 'MD'.

select doctorname from hospital where qualification='MD';

d) Display all doctors who have more than 5 years experience but do not have the qualification 'MD'.

select doctorname from hospital where qualification!='MD' and experience>5;

e) Display the doctors in ‘Skin’ department.

select doctorname from hospital where department=’skin’;

f) update the experience of doctor with doctored=’D003’ to 5.

update hospital set experience = 5 where doctored=’D003’;

g) Delete the doctor with DoctorID=’D005’.

delete from hospital where doctored=’D005’;

Program 4:

Create the following tables Bank_customer (accno primary key, cust_name, place). Deposit (accno foreign key, deposit_no, damount). Loan (accno foreign key loan_no, Lamount).

create table bank_customer(accno varchar(3) primary key,cname varchar(20),place varchar(20));

insert into bank_customer(103,’shithin’,’thmsy’);

insert into bank_customer(102,’vyshakh’,’koyilandy’);

insert into bank_customer(101,’vismaya’,’mukkam’);

insert into bank_customer(104,’aswin’,’areekode’);

insert into bank_customer(105,’anand’,’manjeri’);

create table deposit(accno varchar(3),deposit_no varchar(3),d_amoun tint, foreign key(accno) references bank_customer(accno));

insert into deposit values(103,524,25000);

insert into deposit values(106,525,30000);

insert into deposit values(107,527,18000);

insert into deposit values(104,526,45000);

create table loan(accno varchar(3),loan_no varchar(3),l_amoun tint,foreign key(accno) references bank_customer(accno));

insert into loan values(107,641,37000);

insert into loan values(102,642,28000);

insert into loan values(105,643,32000);

insert into loan values(101,644,29000);

insert into loan values(105,645,43000);

Write the following queries.

a) Display the details of the customers.

select * from bank_customer;

b) Display the customers along with deposit amount who have only deposit with the bank.

```
select cname,d_amount from bank_customer,deposit where bank_customer.accno=deposit.accno;
```

c) Display the customers along with loan amount who have only loan with the bank.

```
select cname,l_amount from bank_customer,loan where bank_customer.accno=loan.accno;
```

d) Display the customers they have both loan and deposit with the bank.

```
select cname from bank_customer,deposit,loan where bank_customer.accno=load.accno and bank_customer.accno=deposit.accno;
```

e) Display the customer who have neither a loan nor a deposit with the bank.

```
select * from bank_customer where accno not in(select accno from deposit union select accno from loan);
```

<u>Program 5:</u>

Create a table employee with fields (EmpID, EName, Salary, Department, and Age). Insert some records. Write SQL queries using aggregate functions and group by clause.

```
create table employee(empid varchar(3) primary key,ename varchar(25),salary int,dept varchar(55),age int);

insert into employee values(103,’arun’,8000,’purchase’,18);

insert into employee values(107,’jithin’,7800,’purchase’,18);

insert into employee values(105,’anupama’,8300,’purchase’,19);

insert into employee values(213,’abay’,11000,’stock’,26);

insert into employee values(237,’akhil’,10800,’stock’,24);
```

A. Display the total number of employees.

```
select from employee count(ename);
```

B. Display the name and age of the oldest employee of each department.

```
select dept,max(age) as maximum_age from employee group by dept;
```

C. Display the average age of employees of each department.

```
select dept,avg(age) as average_age from employee group by dept;
```

D. Display departments and the average salaries.

```
select dept,avg(salary) as average_salary from employee group by dept;
```

E. Display the lowest salary in employee table.

```
select min(salary) as minimum_salary from employee;
```

F. Display the number of employees working in purchase department.

```
select count(ename) as number_of_employees from employee where dept='purchase';
```

G. Display the highest salary in sales department.

```
select max(salary) as highest_salary from employee where dept='sales';
```

H. Display the difference between highest and lowest salary.

```
select max(salary) – min(salary) salary deference from employee;
```

Program 6:

Create a table product with the fields (Product_code primary key, Product_Name, Category, Quantity, Price).

create table product(product_code varchar(3) primary key,product_name varchar(15),catgory varchar(20),quantity int,price int);

Insert some records Write the queries to perform the following.

insert into product values(‘101’,’lexi’,’pen’,27,5);
insert into product values(‘102’,’colgate’,’paste’,8,34);
insert into product values(‘103’,’hamam’,’soap’,40,23);
insert into product values(‘104’,’cello’,’pen’,20,9);
insert into product values(‘105’,’classmate’,’book’,50,42);

a. Display the records in the descending order of Product_Name.

select * from product order by product_name desc;

b. Display Product_Code, Product_Name with price between 20 and 50.

select product_code,product_name from product where price between 20 and 40;

c. Display the details of products which belongs to the categories of ‘bath soap’,‘paste’, or ‘washing powder’.

select product_name from product where category in(‘soap’,’paste’,’pen’);

d. Display the products whose Quantity less than 100 or greater than 500.

select * from product where quantity < 20 or quantity > 100;

e. Display the products whose names starts with 's'.

   select * from product where product_name like ‘s%’;

f. Display the products which not belongs to the category 'paste'.

   select * from product where category not in(‘paste’);

g. Display the products whose second letter is 'u' and belongs to the Category 'washing powder'.

   select product_name from product where product_name like ‘_u%’ and category in (‘washing powder’);

Program 7:

Consider the employee database given below. Give an expression in SQL for each of the following queries:

EMPLOYEE (Employee-Name, City).

create table employee(employee_name varchar(20),city varchar(20));

insert into employee values(‘anupama’,’cochin’);
insert into employee values(‘gayathri’,’pune’);
insert into employee values(‘nubla’,’benguluru’);

WORKS (Employee-Name, Company-Name, Salary).

create table works(employee_name varchar(20),company_name varchar(30),salary int);

insert into works values(‘anupama’,’wipro’,15000);
insert into works values(‘gayathri’,’infosys’,25000);
insert into works values(‘nubla’,’wipro’,22000);

COMPANY (Company-Name, City).

create table company(company_name varchar(30),city varchar(20));

insert into company values(‘wipro’,’benguluru’);
insert into company values(‘infosys’,’benguluru’);

MANAGES (Employee-Name, Manager-Name).

create table manages(employee_name varchar(20),manager_name varchar(20));

insert into manages values(‘anupama’,’diya’);
insert into manages values(‘gayathri’,’yadu’);
insert into manages values(‘nubla’,’noorbina’);

A) Find the names of all employees who work in Infosys.

select employee_name from works where company_name=’infosys’.

B) Find the names and cities of residence of all employees who works in Wipro.

select employee.employee_name,employee.city from employee,works where employee.employee_name=works.employee_name and works.company_name=’wipro’;

C) Find the names, and cities of all employees who work in Infosys and earn more than Rs. 10,000.

select employee.employee_name,employee.city from employee,works where employee.employee_name=works.employee_name and works.company_name='infosys' and salary > 10000;

D) Find the employees who live in the same cities as the companies for which they work.

select employee.employee_name from employee,works,company where company.company.company_name=works.company_name and works.employee_name=employee.employee_name and company_city=employee.city;

E) Find all employees who do not work in Wipro Corporation.

select * from works where company_name not in('wipro');

F) Find the company that has the most employees.

select company_name from works group by company_name having count(distinct employee_name) > = all(select count(distinct employee_name) from works group by company_name);

Program 8:

Write a program code to calculate the area of a circle for a value of radius varying from 3 to 7. Store the radius and the corresponding value of calculated area in an empty table named areas with field's radius and area.

Query

```
create table areas(radius int,area int);


create or replace function calarea() returns void

language plpgsql

as $$

declare

read int:=3;

area int;

begin

loop

area:=3.14*read*read;

insert  into areas values(read,area);

read:=read+1;

exit when read > 7;

end loop;

end;

$$;


select calarea();
select * from areas;
```

Program 9:

Write a program block to calculate the electricity bill by accepting cust_no and units_consumed.

Query

```
ccreate table bill(cons_no int primary key, units int,amount float);


create or replace function elebill(int,int) retruns void

language plpgsql

as $$

declare

cons_no alias for $1;

units alias for $2;

amount float;

begin

amount:=units*6.40;

insert into bill values(cons_no,units,amount);
```

```
end;

$$;


select elebill(123,216);


select * from bill;
```

Program 10:

Create a procedure to print Fibonacci number up to a limit, limit is passed as an argument.

Query

```
create or replace function fibo(int) returns text

langaue plpgsql

as $$

declare

a int:=0;

b int:=1;

c int;

n alias for $1;
```

```
begin

raise notice ‘The fibonacci series is….’;

while a<=n

loop

raise notice ‘%’,a;

c:=a+b;

a:=b;

b:=c;

end loop;

end;

$$;


select fibo(15);
```

<u>Program 11:</u>

Create a function to check whether a given number is prime or not.

<u>Query</u>

```
create or replace function prime(int) returns text
language plpgsql
as $$
```

```
declare
n alias for $1;
i int:=2;
counterint:=1;
msg text;
begin
for i in 2..n/2 loop
if mod(n,i)=0 then
counter:=0;
exit;
end if;
end loop;
if counter = 1 then
msg:=n||' is a prime number';
else
msg:=n||' is not a prime number';
end if;
return msg;
end;
$$;

select prime(31);

select prime(5);

select prime(4);
```

Program 12:

create a table emp_salary(empno, ename,dept,salary). Write a function to return the average salary of a particular department by accepting departmentname as argument.

```
create table emp_salary(empnoint primary key,ename varchar(25),dept varchar(25),salary int);

insert into emp_salary values(100,'riya','sales',10000);
insert into emp_salary values(101,'yadu','sales',13000);
insert into emp_salary values(103,'jithin','production',15000);
insert into emp_salary values(104,'vismaya','production',20000);
insert into emp_salary values(102,'neeraj','hr',18000);

create or replace function avgsal(varchar) returns int
language plpgsql
as $$
declare
dname alias for $1;
avgs int:=0;
begin
select avg(salary) into avgs from emp_salary where dept=dname;
return avgs;
end;
$$;

select avgsal('sales');

select avgsal('production');

select avgsal('hr');
```