1.Consider a restaurant database which contain the following relations. Create the tables with necessary constraints and insert the records as mentioned below. (10)

T1:Customers

cust_id	name	phone
1	Aravind	1234567890
2	Brindha	2345678901
3	Chitra	3456789012
4	Joseph	4567890123
5	Arun	5678901234

T2: Orders

order_id	cust_id	order_date	total_amt
101	1	15-09-2025	******
102	2	16-09-2025	******
103	1	17-09-2025	******
104	3	18-09-2025	******
105	2	15-09-2025	******

T3: MenuItems

item_id	name	price
1	Pizza	92.00
2	Pasta	70.50
3	Salad	50.00
4	Curd rice	40.00
5	Sambar rice	65.00

T4: OrderItems

order_id	item_id	quantity
101	1	2
101	3	1
102	2	2
103	1	1
103	3	1

2. Write the SQL queries for the following: (40)

1. Get the names of customers who placed an order on 15-09-2025. (3)
2. List the average price of the MenuItems. (3)
3. List the number of times a customer placed order along with the customer name. (4)
4. Get the total amount spent by each customer. (4)
5. Find the most ordered item (4)
6. List all the customer names who have placed more than one order. (5)
7. Find customers who have never placed an order. (5)
8. Write a procedure to calculate the total amount of Order table based on the price available in the MenuItems table and quantity available in the OrderItems table. (6)
9. Write a trigger that automatically recalculates and updates the total_amount in the Orders table after a row is inserted into OrderItems. (6)

```
create table Customers(
cust_id int,
name varchar(30) not null,
phone bigint,
constraint cus_pk primary key(cust_id)
);

create table MenuItems(
item_id int,
name varchar(30) not null,
price decimal(10,2),
constraint menu_pk primary key(item_id)
);

create table Orders(
order_id int,
cust_id int,
order_date date ,
total_amt decimal(10,2),
constraint order_pk primary key(order_id),
constraint order_fk foreign key(cust_id) references customers(cust_id)
);

create table Orderitems(
order_id int,
item_id int,
quantity int ,
constraint ord_pk primary key(order_id,item_id),
constraint ord_fk1 foreign key(order_id) references orders(order_id),
constraint ord_fk2 foreign key(item_id) references menuitems(item_id)
);
```

```
insert into Customers values
(1, 'Aravind', '1234567890'),
(2, 'Brinda', '2345678901'),
];   6vc
'/(3, 'Chitra', '3456789012'),
(4, 'Joseph', '4567890123'),
(5, 'Arun', '5678901234');

insert into Orders values
(101, 1, '2025-09-15', NULL),
(102, 2, '2025-09-16', NULL),
(103, 3, '2025-09-17', NULL),
(104, 3, '2025-09-18', NULL),
(105, 2, '2025-09-15', NULL);



insert into MenuItems values
(1, 'Pizza', 92.00),
(2, 'Pasta', 70.00),
(3, 'Salad', 50.00),
(4, 'Curd rice', 40.00),
(5, 'Sambar rice', 65.00);

insert into OrderItems values
(101, 1, 2),
(101, 3, 2),
(102, 2, 1),
(103, 1, 1),
(103, 3, 1);
```

```
--Q!
select c.name from customers c
join orders o
on c.cust_id=o.cust_id
where o.order_date='2025-09-15';
--op
  name
---------
 Aravind
 Brinda
(2 rows)
```

```
--Q2
select round(avg(price),2) as average_price
from MenuItems;
--op
 average_price
---------------
         63.40
(1 row)
```

```
--update
update Orders
set total_amt = subt.calculated_total
from (
select oi.order_id,sum(mi.price * oi.quantity) as calculated_total
from OrderItems oi
join MenuItems mi ON oi.item_id = mi.item_id
group by
oi.order_id
) as subt
where Orders.order_id = subt.order_id;
```

```
 order_id | cust_id | order_date | total_amt
----------+---------+------------+-----------
      104 |       3 | 2025-09-18 |
      105 |       2 | 2025-09-15 |
      101 |       1 | 2025-09-15 |    284.00
      102 |       2 | 2025-09-16 |     70.00
      103 |       3 | 2025-09-17 |    142.00
(5 rows)
```

```
--Q3
select c.name,count(o.order_id) total_orders from customers c
left join orders o
on c.cust_id=o.cust_id
group by c.name;

  name   | total_orders
---------+--------------
 Chitra  |            2
 Brinda  |            2
 Aravind |            1
 Joseph  |            0
 Arun    |            0
(5 rows)
```

```
--Q4
select c.name, sum(o.total_amt) as total_spent
from Customers c
join Orders o on c.cust_id = o.cust_id
group by c.name;
--op
  name   | total_spent
---------+-------------
 Chitra  |      142.00
 Brinda  |       70.00
 Aravind |      284.00
(3 rows)
```

```
--Q5
select m.name from MenuItems m
join OrderItems oi
on m.item_id = oi.item_id
group by m.name
order by sum(oi.quantity) desc
limit 1;
--op
 name
-------
 Pizza
(1 row)
```

```
--Q6
select c.name from Customers c
join Orders o on c.cust_id = o.cust_id
group by c.name
having count(o.order_id) > 1;
--op
  name
--------
 Chitra
 Brinda
(2 rows)
```

```
--Q7
select name from Customers
where cust_id not in (select cust_id from Orders);
--op
  name
--------
 Joseph
 Arun
```

```
--Q8
create or replace procedure update_orders()
language plpgsql
AS $$
begin
update orders o
set total_amt = (
select sum(oi.quantity * m.price) from orderitems oi
join menuitems m
on oi.item_id = m.item_id
where oi.order_id = o.order_id
    );
end;
$$;
call update_orders();
--after update
select * from orders
 order_id | cust_id | order_date | total_amt
----------+---------+------------+-----------
      104 |       3 | 2025-09-18 |
      105 |       2 | 2025-09-15 |
      101 |       1 | 2025-09-15 |    284.00
      102 |       2 | 2025-09-16 |     70.00
      103 |       3 | 2025-09-17 |    142.00
```

```
--Q9
create or replace update_total()
returns trigger
language plpgsql
as $$
begin
update orders o
set total_amt = (
select coalse(SUM(oi.quantity * m.price), 0)
from orderitems oi
join menuitems m
on oi.item_id = m.item_id
where oi.order_id = o.order_id
    )
where o.order_id = NEW.order_id;
return new;
end;
$$;

create trigger trg_update_total
after insert or update or delete
on orderitems
for each row
execute function update_total();

insert into orders values
 (106,3,'2025-09-19',NULL);
--before
 order_id | cust_id | order_date | total_amt
----------+---------+------------+-----------
      104 |       3 | 2025-09-18 |
      105 |       2 | 2025-09-15 |
      101 |       1 | 2025-09-15 |    284.00
      102 |       2 | 2025-09-16 |     70.00
      103 |       3 | 2025-09-17 |    142.00
      106 |       3 | 2025-09-19 |
 insert into orderItems values
 (106,3,2);
--after
 order_id | cust_id | order_date | total_amt
----------+---------+------------+-----------
      104 |       3 | 2025-09-18 |
      105 |       2 | 2025-09-15 |
      101 |       1 | 2025-09-15 |    284.00
      102 |       2 | 2025-09-16 |     70.00
      103 |       3 | 2025-09-17 |    142.00
      106 |       3 | 2025-09-19 |    100.00
```