Capstone assessment solutions

Name: Gaurav Nandgopal Deshmukh

USN NO: 72232433C

Q.1. Create tables and apply constraints.

Ans:

Query Query History

```sql
create table users (
    user_id int primary key,
    name varchar(100) not null,
    email varchar(100) unique not null,
    phone varchar(15) unique,
    location varchar(100) not null
);

insert into users (user_id, name, email, phone, location) values
(1, 'ravi sharma', 'ravi.sharma@gmail.com', '9876543210', 'mumbai'),
(2, 'priya mehta', 'priya.mehta@gmail.com', '9898989898', 'delhi'),
(3, 'amit patel', 'amit.patel@yahoo.com', '8765432109', 'ahmedabad'),
(4, 'sneha rao', 'sneha.rao@gmail.com', '9123456780', 'bangalore'),
(5, 'vikram joshi', 'vikram.joshi@gmail.com', '9012345678', 'pune');
```

Data Output

	user_id [PK] integer	name character varying (100)	email character varying (100)	phone character varying (15)	location character varying (100)
1	1	ravi sharma	ravi.sharma@gmail.com	9876543210	mumbai
2	2	priya mehta	priya.mehta@gmail.com	9898989898	delhi
3	3	amit patel	amit.patel@yahoo.com	8765432109	ahmedabad
4	4	sneha rao	sneha.rao@gmail.com	9123456780	bangalore
5	5	vikram joshi	vikram.joshi@gmail.com	9012345678	pune

Total rows: 5 Query complete 00:00:00.135

Query Query History

```sql
create table restaurants (
    restaurant_id int primary key,
    name varchar(100) not null,
    location varchar(100) not null,
    cuisine varchar(50) not null,
    rating decimal(2,1) check (rating >= 0 and rating <= 5)
);
insert into restaurants (restaurant_id, name, location, cuisine, rating) values
(1, 'pizza palace', 'mumbai', 'italian', 4.3),
(2, 'delhi biryani house', 'delhi', 'indian', 4.1),
(3, 'burger hub', 'bangalore', 'american', 3.9),
(4, 'healthy bowl', 'ahmedabad', 'healthy', 4.4),
(5, 'pasta point', 'pune', 'italian', 4.2);
select * from restaurants;
```

Data Output

	restaurant_id [PK] integer	name character varying (100)	location character varying (100)	cuisine character varying (50)	rating numeric (2,1)
1	1	pizza palace	mumbai	italian	4.3
2	2	delhi biryani house	delhi	indian	4.1
3	3	burger hub	bangalore	american	3.9
4	4	healthy bowl	ahmedabad	healthy	4.4
5	5	pasta point	pune	italian	4.2

Query Query History

```sql
create table menu (
    menu_id int primary key,
    restaurant_id int not null references restaurants(restaurant_id),
    item_name varchar(100) not null,
    price decimal(8,2) check (price >= 0),
    is_available boolean default true
);
insert into menu (menu_id, restaurant_id, item_name, price, is_available) values
(1, 1, 'margherita pizza', 180.00, true),
(2, 1, 'farmhouse pizza', 220.00, true),
(3, 2, 'chicken biryani', 230.00, true),
(4, 2, 'paneer biryani', 200.00, true),
(5, 3, 'cheese burger', 150.00, true),
(6, 4, 'quinoa salad', 180.00, true),
```

Data Output

	menu_id [PK] integer	restaurant_id integer	item_name character varying (100)	price numeric (8,2)	is_available boolean
1	1	1	margherita pizza	180.00	true
2	2	1	farmhouse pizza	220.00	true
3	3	2	chicken biryani	230.00	true
4	4	2	paneer biryani	200.00	true
5	5	3	cheese burger	150.00	true

Total rows: 7 Query complete 00:00:00.161

Query Query History

```sql
create table orders (
    order_id serial primary key,
    user_id int not null references users(user_id),
    restaurant_id int not null references restaurants(restaurant_id),
    order_date date not null,
    status varchar(50) check (status in ('pending', 'delivered', 'cancelled'))
);
insert into orders (order_id, user_id, restaurant_id, order_date, status) values
(1, 1, 1, '2024-10-01', 'delivered'),
(2, 2, 2, '2024-10-02', 'delivered'),
(3, 3, 4, '2024-10-03', 'delivered'),
(4, 4, 3, '2024-10-04', 'cancelled'),
(5, 5, 5, '2024-10-05', 'delivered');
select * from orders;
```

Data Output

	order_id [PK] integer	user_id integer	restaurant_id integer	order_date date	status character varying (50)
1	1	1	1	2024-10-01	delivered
2	2	2	2	2024-10-02	delivered
3	3	3	4	2024-10-03	delivered
4	4	4	3	2024-10-04	cancelled
5	5	5	5	2024-10-05	delivered

Total rows: 5 Query complete 00:00:00.166

Query Query History

```sql
create table order_items (
    order_item_id int primary key,
    order_id int not null references orders(order_id),
    menu_id int not null references menu(menu_id),
    quantity int check (quantity > 0)
);
insert into order_items (order_item_id, order_id, menu_id, quantity) values
(1, 1, 1, 1),
(2, 1, 2, 1),
(3, 2, 3, 1),
(4, 2, 4, 1),
(5, 3, 6, 1),
(6, 4, 5, 2),
(7, 5, 7, 1);
select * from order_items;
```

Data Output

	order_item_id [PK] integer	order_id integer	menu_id integer	quantity integer
1	1	1	1	1
2	2	1	2	1
3	3	2	3	1
4	4	2	4	1
5	5	3	6	1

Total rows: 7 Query complete 00:00:00.250

Query Query History

```sql
create table reviews (
    review_id int primary key,
    user_id int not null references users(user_id),
    restaurant_id int not null references restaurants(restaurant_id),
    rating decimal(2,1) check (rating >= 0 and rating <= 5),
    review_text text,
    review_date date not null
);
insert into reviews (review_id, user_id, restaurant_id, rating, review_text, review_date) values
(1, 1, 1, 4.5, 'pizza was fresh and cheesy', '2024-10-01'),
(2, 2, 2, 4.2, 'biryani tasted great', '2024-10-02'),
(3, 3, 4, 4.4, 'healthy and light meal', '2024-10-03'),
(4, 5, 5, 4.3, 'creamy and well-cooked pasta', '2024-10-05');
select * from reviews;
```

Data Output

Showing ro

	review_id [PK] integer	user_id integer	restaurant_id integer	rating numeric (2,1)	review_text text	review_date date
1	1	1	1	4.5	pizza was fresh and cheesy	2024-10-01
2	2	2	2	4.2	biryani tasted great	2024-10-02
3	3	3	4	4.4	healthy and light meal	2024-10-03
4	4	5	5	4.3	creamy and well-cooked pasta	2024-10-05

Q.2. List all restaurants along with their location and cuisine.

Ans:

```sql
select name, location, cuisine
        from restaurants;
```

Data Output

	name character varying (100)	location character varying (100)	cuisine character varying (50)
1	pizza palace	mumbai	italian
2	delhi biryani house	delhi	indian
3	burger hub	bangalore	american
4	healthy bowl	ahmedabad	healthy
5	pasta point	pune	italian

Total rows: 5 Query complete 00:00:00.166

Q.3. List all menu items with their prices and availability for the restaurant named "Pizza Palace".

Ans:

```
select m.item_name,
        m.price,
        m.is_available from menu m join restaurants r on m.restaurant_id = r.restaurant_id
        where r.name = 'pizza palace';
```

Data Output

	item_name character varying (100)	price numeric (8,2)	is_available boolean
1	margherita pizza	180.00	true
2	farmhouse pizza	220.00	true

Q.4. Display the total number of users in each location.

Ans:

```
select location,
        count(*) as total_users from users
        group by location;
```

Data Output

	location character varying (100)	total_users bigint
1	mumbai	1
2	bangalore	1
3	delhi	1
4	pune	1
5	ahmedabad	1

Total rows: 5 | Query complete 00:00:00.156

Q.5. List all orders with user name, restaurant name, number of items in the order.

Ans:

```
select o.order_id,
        u.name as user_name,
        r.name as restaurant_name,
        count(oi.order_item_id) as total_items from orders o join users u on o.user_id =
        u.user_id
        join restaurants r on o.restaurant_id = r.restaurant_id
        join order_items oi on o.order_id = oi.order_id group by o.order_id, u.name, r.name;
```

Data Output

	order_id integer	user_name character varying (100)	restaurant_name character varying (100)	total_items bigint
1	3	amit patel	healthy bowl	1
2	5	vikram joshi	pasta point	1
3	2	priya mehta	delhi biryani house	2
4	1	ravi sharma	pizza palace	2
5	4	sneha rao	burger hub	1

Total rows: 5 Query complete 00:00:00.139

Q.6. Find the average rating of each restaurant and list them in descending order.

Ans:

```
select r.name as restaurant_name,
        (avg(rv.rating), 0) as average_rating from restaurants r
        left join reviews rv on r.restaurant_id = rv.restaurant_id
        group by r.name order by average_rating desc;
```

Data Output

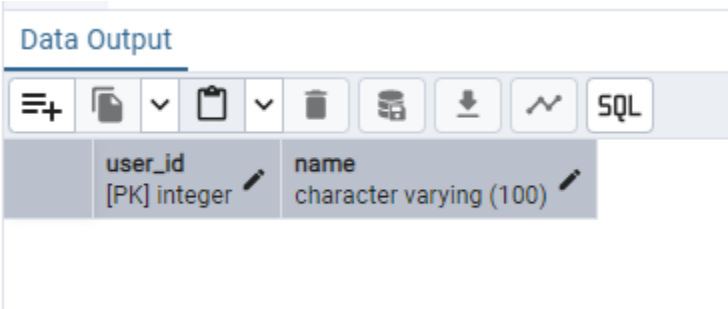
	restaurant_name character varying (100)	average_rating numeric
1	pizza palace	4.5000000000000000
2	healthy bowl	4.4000000000000000
3	pasta point	4.3000000000000000
4	delhi biryani house	4.2000000000000000
5	burger hub	0

Total rows: 5 Query complete 00:00:00.156

Q.7. Find the users who have never placed an order.

Ans:

```
select u.user_id,
u.name from users u
left join orders o on u.user_id = o.user_id where o.order_id is null;
```



Q.8. Update all menu items by increasing their price by 10% where the current price is below 200.

Ans:

```
update menu
set price = price * 1.10
where price < 200;
select * from menu;
```

Data Output

	menu_id [PK] integer	restaurant_id integer	item_name character varying (100)	price numeric (8,2)	is_available boolean
1	2	1	farmhouse pizza	220.00	true
2	3	2	chicken biryani	230.00	true
3	4	2	paneer biryani	200.00	true
4	7	5	penne alfredo	210.00	true
5	1	1	margherita pizza	198.00	true
6	5	3	cheese burger	165.00	true
7	6	4	quinoa salad	198.00	true

Total rows: 7 Query complete 00:00:00.132

Q.9. Find all restaurants that serve more menu items than the average number of items across all restaurants.

Ans:

```
select r.name,
        count(m.menu_id) as item_count from restaurants r
        join menu m on r.restaurant_id = m.restaurant_id
        group by r.restaurant_id,
        r.name having count(m.menu_id) > ( select avg(item_count)
        from ( select count(menu_id) as item_count
        from menu group by restaurant_id ) as avg_items
        );
```

Data Output

	name character varying (100)	item_count bigint
1	delhi biryani house	2
2	pizza palace	2

Q.10. Get the top 3 most ordered items across all restaurants.

Ans:

```
select m.item_name,
        sum(oi.quantity) as total_ordered from order_items oi
        join menu m on oi.menu_id = m.menu_id
        group by m.item_name
        order by total_ordered desc limit 3;
```

Data Output

	item_name character varying (100)	total_ordered bigint
1	cheese burger	2
2	penne alfredo	1
3	margherita pizza	1

Q.11. Create a function get_total_order_amount(order_id) that calculates and returns the total amount.

Ans:

```
create or replace function get_total_order_amount(p_order_id int)
        returns numeric as $$
        declare
                total_amount numeric;
        begin
                select sum(m.price * oi.quantity)
                into total_amount
                from order_items oi
                join menu m on oi.menu_id = m.menu_id
                where oi.order_id = p_order_id;
        return coalesce(total_amount, 0);

        end;
         $$ language plpgsql;
        select get_total_order_amount(2);
```

Data Output

	get_total_order_amount numeric
1	430.00

Q.12. Write a stored procedure place_order that accepts a user id, restaurant id, and a list of item ids with quantities and inserts records into the Orders and Order_items tables.

Ans:

```
create or replace procedure place_order(

        p_user_id int, p_restaurant_id int,

        p_item_ids int[], p_quantities int[] )

        language plpgsql

        as $$

        declare new_order_id int; i int; begin

         insert into orders(user_id, restaurant_id, order_date, status)

        values (p_user_id, p_restaurant_id, current_date, 'pending') returning order_id into
        new_order_id;


        for i in array_lower(p_item_ids, 1)..array_upper(p_item_ids, 1)
        loop
        insert into order_items(order_id, menu_id, quantity)
        values (new_order_id, p_item_ids[i], p_quantities[i]);
        end loop;


        end; $$;

        call place_order(

        1, -- user_id

        1, -- restaurant_id

        array[1, 2], -- menu item ids

        array[2, 1] -- corresponding quantities

        );
```

select * from orders;

Data Output

	order_id [PK] integer	user_id integer	restaurant_id integer	order_date date	status character varying (50)
1	1	1	1	2024-10-01	delivered
2	2	2	2	2024-10-02	delivered
3	3	3	4	2024-10-03	delivered
4	4	4	3	2024-10-04	cancelled
5	5	5	5	2024-10-05	delivered

Q.13. Create a trigger that updates the restaurant's average rating in the Restaurants table whenever a new review is inserted into the Reviews table.

Ans:

```
create or replace function update_restaurant_rating()
        returns trigger as $$
        begin
                update restaurants
                set rating = (
                        select avg(rating)
                        from reviews
                        where restaurant_id = new.restaurant_id )
                where restaurant_id = new.restaurant_id;
return new;

end;
```

```
$$ language plpgsql;


create trigger trg_update_rating

after insert on reviews

for each row

execute procedure update_restaurant_rating();

insert into reviews (review_id, user_id, restaurant_id, rating, review_text, review_date)
values (5, 1, 1, 4.9, 'best pizza ever', current_date);

select * from restaurants where restaurant_id = 1;
```

Data Output

	restaurant_id [PK] integer	name character varying (100)	location character varying (100)	cuisine character varying (50)	rating numeric (2,1)
1	1	pizza palace	mumbai	italian	4.7