Assignment 5

Instructions

- Write and execute the commands and paste the screenshots of the query as well as the output.
- Submission should be a dedicated PDF file containing all the answers. Nomenclate the PDF as <your_roll_no>_assignment5.pdf
- The code should be executable and the output should be clearly visible.
- Leverage the DVDRental Database used in assignment 2 for solving the questions below.

Data Types

1. **Update the address of the customer** to now hold their exact coordinates (**latitude** and **longitude**) with precision up to 18 digits and the decimal part scaling over 15 digits. Insert at least one record with address ID = '123456' into the modified address and display (only) this record.
2. **Update the payment details of the customer** to now hold the **transaction id** corresponding to each payment that occurred. The transaction id is a

32-digit unique identifier that is automatically generated by the computer. Display at least 5 payment records after adding the transaction ID.

3. **Update the payment details of the customer** to now hold the **payment info** corresponding to each payment that occurred. The payment info is a nested JSON string. Copy and insert the following JSON data after modifying the payment details and display the same -

```
{ "intent": "sale", "payer": { "payment_method": "paypal" }, "redirect_urls": { "return_url":
"http://return.url", "cancel_url": "http://cancel.url" }, "transactions": [{ "item_list": {
"items": [{ "name": "item", "sku": "item", "price": "1.00", "currency": "USD", "quantity": "1" }]
}, "amount": { "currency": "USD", "total": "1.00" }, "description": "This is the payment
information." }] }
```

Users, Roles, and Authorization

---

1. Add a new user **'Sergio'** who has the role of an **owner** and can create new roles for new/existing users in the database.
    a. Display all active users and their roles.
    b. Restart the psql session as user 'Sergio' and try creating a database. Share the screenshot of the output.
2. (Logged-in as 'Sergio' only) Create the role of a **'Receptionist'** who manages only the rental data, the customer data, and the payment details.

A receptionist can perform any data modification over the aforementioned data (fetch / add / delete / update records).

   a. Display all privileges for the 'Receptionist' over the specified data scope.
   b. Restart the psql session as user 'Receptionist' and try displaying all staff records using its appropriate relation. Share the screenshot of the output.

NOTE: Restart the session as a superuser (**postgre**) to attempt questions 3 & 4.

3. Create a new user **'DB Administrator'** having complete access to the entire database. Log in as the administrator and modify the roles of the owner so that the owner now has all the privileges, not just creating roles. Display all the modified privileges for the owner.
4. Create a new role of a **'Manager'** and assign a new user **'Max'** as the manager who requires a login to access the records. Create another role **'Logged_in'** which keeps track of all the active users. Let 'Max' be logged in, thereby, is assigned the respective role.
   a. Display all the users currently active in the database.
   b. Remove the role of a manager completely but do not remove the employees who were managers previously. Share the screenshot of the status of the users in the system.

Functions and Procedures

1. Create a function that consumes an **‘ID’** number as an argument and **returns the number of films** rented by the customer corresponding to that particular ID.
2. Create a function that consumes an **‘ID’** number as an argument and **returns the complete details of the customer** corresponding to that particular ID ranging from the customer name (full), their address ID, and the amount paid by them including the payment date.
3. Create a function that consumes **‘actor ID’** as an argument and **returns all the films the actor corresponding to the input actor ID has acted in**. The function output should be a well-organized table enveloping the film ID, film title, and the complete name (first & last) of the actor.
4. Create a procedure that **displays all the film IDs** in ascending order and the name (title) of each film corresponding to that particular ID.

E.O.D.