Normalisation and CTE queries

Assignment Questions

PW SKILLS

- **First Normal Form (1NF):**
  - Identify a table in the Sakila database that violates 1NF. Explain how you would normalize it to achieve 1NF.
- **Second Normal Form (2NF):**
  - Choose a table in Sakila and describe how you would determine whether it is in 2NF. If it violates 2NF, explain the steps to normalize it.
- **Third Normal Form (3NF):**
  - Identify a table in Sakila that violates 3NF. Describe the transitive dependencies present and outline the steps to normalize the table to 3NF.
- **Normalization Process:**
  - Take a specific table in Sakila and guide through the process of normalizing it from the initial unnormalized form up to at least 2NF.
- **CTE Basics:**
  - Write a query using a CTE to retrieve the distinct list of actor names and the number of films they have acted in from the `actor` and `film_actor` tables.
- **Recursive CTE:**
  - Use a recursive CTE to generate a hierarchical list of categories and their subcategories from the `category` table in Sakila.
- **CTE with Joins:**
  - Create a CTE that combines information from the `film` and `language` tables to display the film title, language name, and rental rate.
- **CTE for Aggregation:**
  - Write a query using a CTE to find the total revenue generated by each customer (sum of payments) from the `customer` and `payment` tables.
- **CTE with Window Functions:**
  - Utilize a CTE with a window function to rank films based on their rental duration from the `film` table.
- **CTE and Filtering:**
  - Create a CTE to list customers who have made more than two rentals, and then join this CTE with the `customer` table to retrieve additional customer details.
- **CTE for Date Calculations:**
  - Write a query using a CTE to find the total number of rentals made each month, considering the `rental_date` from the `rental` table.
- **CTE for Pivot Operations:**
  - Use a CTE to pivot the data from the `payment` table to display the total payments made by each customer in separate columns for different payment methods.
- **CTE and Self-Join:**
  - Create a CTE to generate a report showing pairs of actors who have appeared in the same film together, using the `film_actor` table.
- **CTE for Recursive Search:**
  - Implement a recursive CTE to find all employees in the staff table who report to a specific manager, considering the `reports_to` column.