Project #7: Summary and Subqueries

Use the Sakila database for this project.

1 Summary Queries (1 point each x 5 = 5 points)

Write summary queries to answer the following questions. Note how many columns should be in the output (use numbers in parentheses as a guide). In each query, provide friendly column names in the results list. Before each SQL command, provide a comment giving the question number and how many rows resulted from the query.

1. The Rental table contains movie rental data. Get a list that shows how many rentals occurred on each date. The results should show (1) the date and (2) the rental count. Show only those days where over 100 rentals occurred. Hint: you'll need to isolate the date portion; explore functions like LEFT and DATE.

2. The Rental table shows what staff member (by id) rented the film. The Staff table contains detailed staff data. Get a list of (1) the name of each staff member (in "last, first" format) along with (1) each month and year they rented films, plus (3) how many films they rented that month. Sort by the staff member, and, within that, the rental count (highest first). Hint: you'll need to isolate the month and year; explore functions like MONTH and YEAR.

3. The Payments table contains customer payment data. Get a total of payments for each month of each year, showing in the results (1) the year, (2) month, and (3) payment total. Sort the list so that the most recent month is at the top and the oldest one is at the bottom.

4. Using data from Payment and the customer data tables (Customer, Address), get a list of (1) the customer name ("last, first" format), (2) district in which they live, along with (3) the count of payments and (4) the total of payments they have made. Show only the customers who have payment totals of $150 or over. Sort the list by customer name.

5. Use the Film and Category tables (along with any others that may be necessary) to get a count of films in each category. In the results show (1) the film count (highest first) and (2) the category (e.g., "Documentary"). Show only categories with more than 70 films in them.

2 Write Subqueries (1 point each x 5 = 5 points)

For each of these questions, write a SELECT query that contains a subquery. Before each SQL command, provide a comment giving the question number and how many rows resulted from the query.

6. The Country table contains a country ID and name. In that table, locate the IDs of countries that have the word "united" anywhere in their name. Then, in the City table, locate and list cities (by name, not number) in those countries, but only the ones with the word "south" anywhere in the city's name. Sort the results by the city's name.

7. Use the Payment and Customer tables to get a list of customers (show the customer ID, first and last names, and email address) who have ever made payments of $10.00 or more.

8. Write a subquery that finds film IDs (from the Film table) greater than 180 minutes in length. Then use that to help find a list of category names (from the Category table) that have associated films of that length. Don't show repeated category names.

9. A customer calls and wants to rent a copy of "Airplane Sierra". Get a list of cities (by name, not ID, and with no repeats) where they can find that film in inventory. Start by creating a subquery that finds the film ID for the film.

10. Get a list of film category IDs for categories that have over sixty films in them. Use that result to get a list of the actors who are in over twenty-five of the movies in those categories. In the result, show the actor's last name, first name, and the count of films they are in.

3 Submitting Your Work

Submit your SQL file in Canvas.

4 Hints

- Some of these are challenging; you'll need to think before you code, here, perhaps more than you've had to on most other exercises.
- JOINs are permitted either in the inner or outer query; you shouldn't need more than one inner and one outer query for any problem in this set.
- For the subquery problems, follow the workflow I showed you in class.
- For inner subqueries, heed my warning about returning too much data. Be wary of cases where you're tempted to return multiple columns from an inner query; that's usually a plan that will cause you too much work in the outer query. Also, be wary of returning non-PK/non-FK data from subqueries; we usually want lookup data for use in the outer query, and keys are usually the way that will happen.

5 Grading

Projects are worth ten points. You'll be graded on correctness (does your query answer the question correctly?), approach (does your query use the technology it's supposed to use?) and style (does it follow our style guide and is it readable?).