SQL Intermediate Project Guide

This project is designed to help you practice and demonstrate your SQL skills at an intermediate level. You'll work with a realistic database (e.g., a movie ratings dataset, a sales database, or a customer order system) and apply core SQL concepts such as joins, aggregation, subqueries, CTEs, window functions, and optimization techniques.

Project Objectives

- Understand the schema of a relational database.
- Write complex queries involving multiple joins.
- Use aggregation, grouping, filtering (HAVING).
- Apply subqueries and Common Table Expressions (CTEs).
- Use window functions for analytical queries.
- Optimize queries using indexes and EXPLAIN plans (optional).

Step-by-Step Instructions

1. Understand the Dataset

```
Download a relational dataset from Kaggle (e.g., Northwind Traders, IMDB Movies, Chinook DB). Import it into a SQL database (PostgreSQL, MySQL, or SQLite). Explore the tables, keys, and relationships.
```

2. Schema Exploration

```
Write queries to: - List all tables and their columns - Identify primary and foreign keys - Count rows in each table
```

3. Basic Queries

```
Write SELECT queries with filtering, ordering, and aliases. Example: ```sql SELECT customer_name, order_date FROM orders WHERE order_date >= '2024-01-01'; ```
```

4. Joins Practice

```
Perform INNER JOIN, LEFT JOIN, and RIGHT JOIN across 2–3 tables. Example: ```sql SELECT c.name, o.order_date, p.product_name FROM customers c JOIN orders o ON c.customer_id = o.customer_id JOIN products p ON o.product_id = p.product_id; ```
```

5. Aggregation & Grouping

```
Use COUNT, SUM, AVG, MAX, GROUP BY, HAVING. Example: ```sql SELECT customer_id, COUNT(*) AS total_orders FROM orders GROUP BY customer_id HAVING COUNT(*) > 5; ```
```

6. Subqueries and CTEs

```
Write correlated and uncorrelated subqueries. Example: ```sql SELECT name FROM customers WHERE id IN (SELECT customer_id FROM orders WHERE total > 100); ``` Use CTEs for better structure: ```sql WITH high_value_orders AS ( SELECT * FROM orders WHERE total > 100 ) SELECT * FROM customers WHERE id IN (SELECT customer_id FROM high_value_orders); ```
```

7. Window Functions

Use RANK(), ROW_NUMBER(), LAG(), LEAD() for advanced analytics. Example: ```sql SELECT customer_id, order_date, RANK() OVER (PARTITION BY customer_id ORDER BY order_date) AS order_rank FROM orders; ```

8. Query Optimization (Optional)

Use EXPLAIN or EXPLAIN ANALYZE to evaluate performance. Consider creating indexes on high-frequency filtering columns.

9. Bonus Challenge

Create a dashboard-style summary using only SQL (e.g., Top 5 customers by revenue, Monthly sales trends, Most popular products).