# ASSIGNMENT 1: SQL MASTERY - The E-Commerce Analytics Challenge

**Topics Covered/focused**: SQL Manipulation, Retrieval, Aggregation, Joins, Subqueries, CTEs, Window Functions, Stored Procedures, Indexes

**Duration**: 7-8 hours | **Questions****: 30

**Problem Statement**

You're hired as a Data Engineer at **"ShopHub"** - a fast-growing e-commerce platform. The company has raw transactional data scattered across multiple systems. Your job is to design, optimize, and create a complete analytics database.

**Dataset**

**Brazilian E-Commerce Public Dataset by Olist**

-> **Link****: https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce

- **> Size****: 100K orders, 8 CSV files (~45 MB)

- **> Files****: customers, orders, order_items, products, sellers, order_payments, order_reviews, geolocation

- > **Features****: Real data with nulls, duplicates, multi-table relationships

- **> Alternative****: https://github.com/olist/brazilian-ecommerce (same dataset)

**Hints/ insights about dataset :**

a) Real production-quality data with actual issues

b) Multiple tables requiring complex joins

c) Has data quality problems to fix

d) Good for aggregations and window functions

e) Free and publicly available

**Assignment Questions (30 Total):**

**#### Part A: Database Design & Data Quality (5 questions)**

1. Design a normalized database schema (3NF) with ERD showing all relationships between 8 tables

2. Identify and document 10+ data quality issues in the raw CSV files (nulls, duplicates, format inconsistencies, orphan records)

3. Create tables with appropriate data types, primary keys, foreign keys, and constraints (NOT NULL, CHECK, UNIQUE)

4. Write a SQL script to detect and remove duplicate customers while keeping the most recent record

5. Create a comprehensive data dictionary documenting each table, column, data type, and business meaning

**#### Part B: Data Manipulation & Retrieval (5 questions)**

6. Insert cleaned data using transactions - demonstrate COMMIT and ROLLBACK with error scenarios

7. Find customers who have the same email but different names/addresses (data quality red flag)

8. Identify orphan records: order_items that reference non-existent products or orders

9. List customers who registered but never placed an order (conversion funnel leak analysis)

10. Detect potential fraud: orders where payment_value != order total (sum of order_items)

**#### Part C: Complex Aggregations (5 questions)**

11. Calculate monthly revenue with Month-over-Month (MoM) growth percentage - handle first month edge case

12. Find top 10 products by revenue in each category using RANK() or DENSE_RANK()

13. Calculate Customer Lifetime Value (CLV) = SUM(payment_value) per customer, categorize as Bronze/Silver/Gold

14. Create a sales report with daily, weekly, monthly subtotals using ROLLUP or CUBE

15. Identify seasonal patterns: which product categories sell best in which months?

**#### Part D: Mastering Joins (5 questions)**

16. Create a complete customer 360-degree view joining 5+ tables (customers, orders, order_items, products, reviews)

17. Find customers who bought products from category 'electronics' but never from 'books'

18. List sellers and their best-selling product in each category they sell

19. Identify products frequently bought together (market basket analysis using self-join)

20. Find orders with shipping delays: expected_delivery_date < actual_delivery_date, show customer and seller info

**#### Part E: Subqueries & CTEs (4 questions)**

21. Find customers who spent more than the average customer in their state (correlated subquery)

22. Get the 2nd highest revenue-generating product in each category (ranking with subquery)

23. Using recursive CTE, create a product category hierarchy (if categories have parent-child relationships)

24. Find customers who made purchases in 3+ consecutive months using CTE with window functions

**#### Part F: Advanced Window Functions (4 questions)**

25. Calculate 7-day moving average of daily order count

26. For each customer, find the gap (in days) between consecutive orders using LAG()

27. Rank sellers by revenue within each state, show only top 3 per state

28. Calculate running total of revenue by date with percentage of grand total

**#### Part G: Stored Procedures & Optimization (2 questions)**

29. Create a stored procedure to calculate dynamic discount based on rules:

    - New customer (first order): 15% off

    - Order value > $500: 10% off

    - Loyal customer (5+ previous orders): 5% off

    - Apply maximum one discount per order

30. Identify 3 slowest queries using EXPLAIN ANALYZE, optimize with appropriate indexes (B-tree, Hash), show before/after execution time

**Deliverables**

- SQL script files organized in folders (01_schema, 02_data_cleaning, 03_queries, 04_optimization)
- ERD diagram (draw.io/Lucidchart/dbdiagram.io)
- Data quality findings report (PDF/Markdown)
- Performance optimization report with EXPLAIN outputs
- 5-minute video walkthrough

*

*********