**Project: Online Store SQL Management System**

**Overview**

In this challenge, you’ll demonstrate your ability to design and build a fully-functional **SQL database system** simulating an **online store**. You will define the schema, generate realistic data, and use **advanced SQL techniques** to analyze and optimize performance.

This challenge will test your skills in **data modeling**, **SQL scripting**, **query logic**, and **performance optimization**—similar to what real businesses expect from junior data engineers or analysts.

💡 **Important Note**: You must **generate your own sample data** using tools like Mockaroo, Python, or Excel or with the help of Ai tools. Your final output should reflect **realistic store data** (products, customers, payments, etc.)

**Scenario**

You’ve been hired by a retail tech startup to create the backend SQL database of a brand-new **Online Store Platform**. The platform will manage:

* Customer accounts and purchase history
* Products, stock levels, and prices
* Orders and shipping details
* Payment status and transaction logs
* Customer reviews and product ratings

The management team has not provided rigid requirements. Instead, you are expected to **design the data model**, **simulate realistic data**, and build the logic that enables full store operations and insights.

**Your Tasks:**Note: You need to generate the data in Excel, TXT, or CSV files and then transform it into SQL.

1. Data Generation:

* Use Ai help to data generation**, Mockaroo, Python, or similar** to generate **6+ CSV or Excel files**:
  + customers.csv, products.csv, orders.csv, payments.csv, reviews.csv, shipments.csv and etc…
* Each file must have at least:
  + **1000+ customers**, 100**+ products**, **50000+ orders**, and **10-20+ reviews per product (**the more data generation the better your project with realistic data**)**

**2. Data Transformation & Advanced SQL Development:**

* Load the raw CSV files into **staging tables**
* Clean and validate data via SQL transformations
* Insert cleaned data into final normalized tables using **INSERT INTO ... SELECT** logic
* Track transformation history (timestamps, record count logs)
* Create at least:
  + **3 Views** (e.g., Sales Summary, Product Popularity, Customer Lifetime Value)
  + **2 Stored Procedures** (Below you have 4. stored procs and automation, beside that you can create your own logic. e.g., Place Order, Update Inventory.)
  + **1 Trigger** (e.g., stock update or logging change) – optional but recommended for learning purposes
  + **3-5+ Complex Queries** (JOINs, Subqueries, Aggregations)

(Note, you can create more, or you can add your additional logics)

**3. Schema Design:**

* Design a normalized schema (up to **3NF**)
* Include:
  + Primary & foreign keys, constraints…
  + Audit columns (created\_at, updated\_at)
* Share your **Database Diagram**

**4. Stored Procedures & Automation:**

* Write stored procedures to:
  + Load data from staging to final tables
  + Update inventory after each order
  + Log failed inserts or duplicates
* Automate your procedures via:
  + **SQL Server Agent**, **PostgreSQL cron job**, or **task scheduler or etc…**
  + Schedule daily/weekly refresh from updated CSV files

**5. Making documentation:**

* Make a documentation after completing the project (research how to documented your work)

**Submission Guidelines**

* ✅ .sql scripts for schema + data
* ✅ .pdf or .docx report (it is documentation)
* ✅ Screenshots or .png images of visual outputs (Like database diagrams)
* Please add Statement of originality to your documentation.

**Evaluation Criteria**

* **Generated data:** How you are able to generated data and how realistic it was created?
* **Critical Thinking**: How well did you interpret the business problem and address it with data?
* **Technical Execution**: Quality of your data transformations and metric definitions.
* **Clarity**: Are your insights and recommendations well-articulated and actionable?
* **Creativity**: Innovative approaches to solving the problem or presenting data

**✅ Bonus Tips**

* Include anomalies like:
  + Out-of-stock products
  + Delayed shipments
  + Negative reviews or returns
* Consider implementing **basic analytics** such as:
  + Revenue trends
  + Best-selling products
  + Most valuable customers