**Inventory & Warehouse Management System**

**1. Introduction**

This project simulates a real-world inventory and warehouse management system using SQL. The goal is to build a backend database that can track multiple products across different warehouses and handle supplier data, stock updates, alerts for low stock, and inter-warehouse transfers.

It is part of my Elevate Labs journey, focused on developing SQL expertise through hands-on learning.

**2. Objectives**

* Design a normalized relational database using MySQL.
* Simulate real-time inventory operations such as stock tracking and restocking.
* Use advanced SQL features like Triggers and Stored Procedures.
* Practice schema documentation, data integrity, and ER diagram creation.

**3. Tools & Technologies Used**

* **MySQL Workbench / DBeaver** – Database Design & Execution
* **dbdiagram.io** – ER Diagram Creation

**4. Database Schema Design**

**4.1 Tables**

* **Products** – product details including reorder level and supplier ID.
* **Suppliers** – contact info of suppliers.
* **Warehouses** – storage locations with capacity and name.
* **Stock** – bridge table that connects products with warehouses and quantity.
* **Low Stock Alerts** – triggered automatically when quantity drops below reorder level.

**4.2 Relationships**

* Each product is associated with a supplier.
* Each warehouse can store many products.
* Alerts are generated based on reorder levels.
* Stock is tracked per product per warehouse.

**4.3 ER Diagram**

A computer screen shot of a computer

AI-generated content may be incorrect.

**💻 5. SQL Features Implemented**

**5.1 DDL (Data Definition Language)**

* Created tables with PRIMARY and FOREIGN KEY constraints.
* Applied NOT NULL, AUTO\_INCREMENT, and DEFAULT settings.

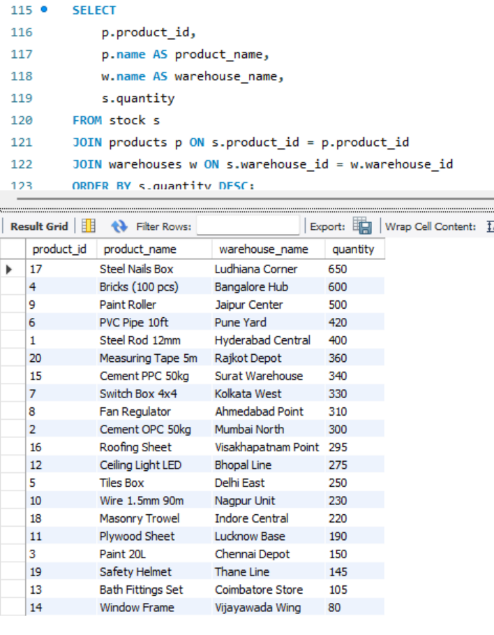
**5.2 DML (Data Manipulation Language)**

* Inserted 20+ sample rows per table.
* Updated data using realistic product names and quantities.

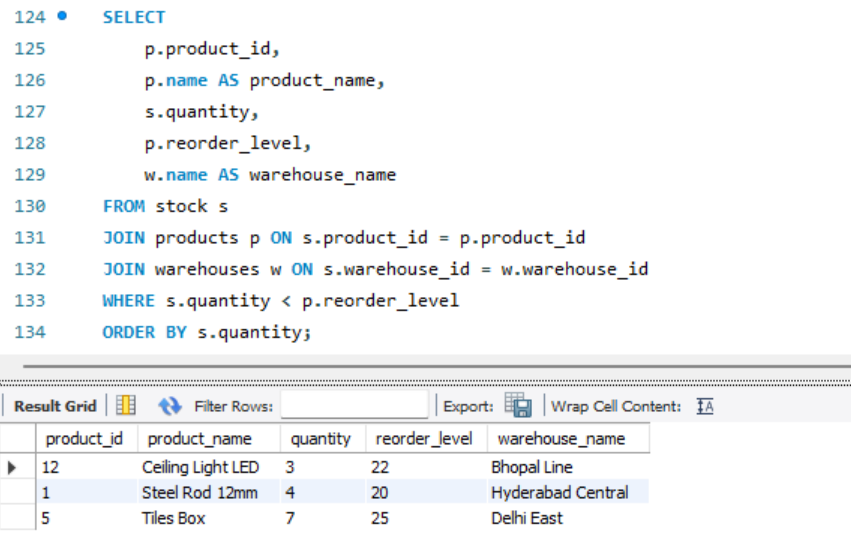
**5.3 JOIN Queries**

* Used INNER JOIN to combine product, warehouse, and stock data.
* Retrieved low-stock items across warehouses.

1. Show Current Stock Levels by Product & Warehouse



1. Reorder Alert – Products Below Reorder Level



3. Warehouse Capacity Usage (Total Stock Units vs Capacity)

A screenshot of a data

AI-generated content may be incorrect.

1. **Total Stock Value Per Warehouse**

A screenshot of a computer

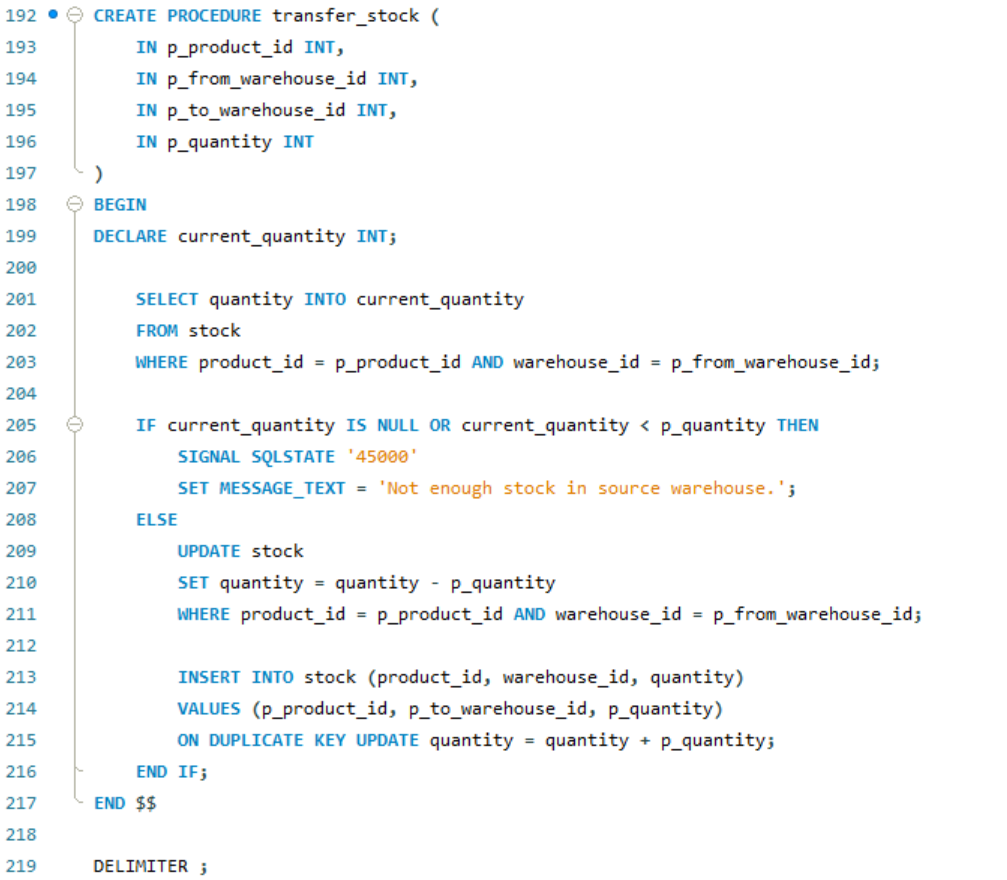
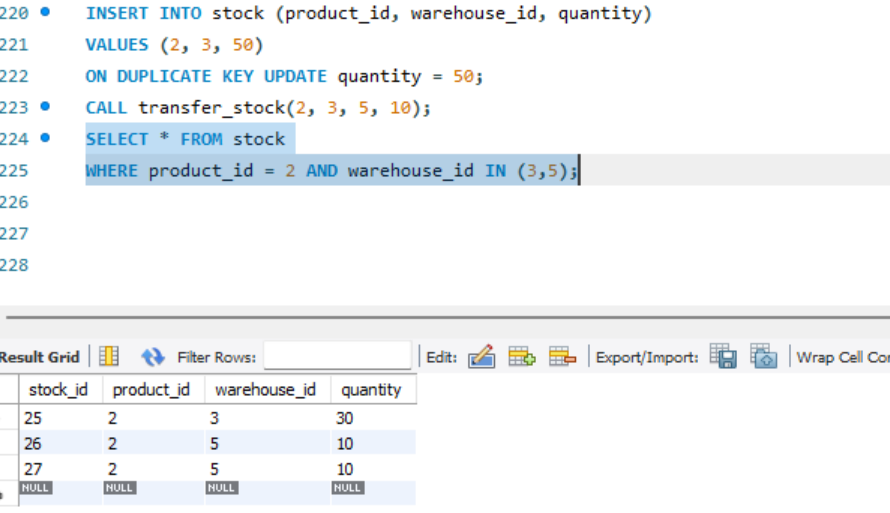
AI-generated content may be incorrect.

**5.4 Triggers**

* Automatically inserted rows into low\_stock\_alerts when quantity < reorder level.
* A screenshot of a computer

  AI-generated content may be incorrect.

**5.5 Stored Procedures**

* Created transfer\_stock() to simulate moving stock from one warehouse to another.
* 
* 

**6. Learning Outcomes**

* Deepened understanding of relational databases and normalization (1NF–3NF)
* Hands-on experience with triggers and procedures
* Improved problem-solving through business logic simulation
* Strengthened backend design and query writing confidence

**7. Acknowledgements**

A big thank you to **Elevate Labs** for offering a structured environment that motivated me to apply SQL in real-world projects. This guided journey gave me the discipline and direction to build something meaningful and practical.

Bottom of Form