**Inventory Management System Project Report**

**Project Overview**

The Inventory Management System is a relational database designed to streamline the management of inventory, suppliers, purchase orders, and sales for a business. The database, named InventoryDB, is implemented using SQL and consists of five core tables: PRODUCTS, SUPPLIERS, PURCHASE\_ORDERS, PURCHASE\_ORDER\_ITEMS, and SALES. This system enables efficient tracking of stock levels, supplier interactions, procurement, and sales activities, with built-in queries to generate actionable insights.

**Database Schema**

The database schema is structured to maintain data integrity and support business operations. Below is a summary of the tables and their relationships:

**Tables**

1. **PRODUCTS**
   * Stores product details such as product\_id (primary key), name, category, price, stock\_quantity, and reorder\_level.
   * Used to manage inventory items and monitor stock levels.
2. **SUPPLIERS**
   * Contains supplier information: supplier\_id (primary key), name, and contact\_info.
   * Tracks vendors supplying products.
3. **PURCHASE\_ORDERS**
   * Records purchase orders with order\_id (primary key), supplier\_id (foreign key referencing SUPPLIERS), order\_date, and total\_amount.
   * Manages procurement from suppliers.
4. **PURCHASE\_ORDER\_ITEMS**
   * Details items in each purchase order: order\_item\_id (primary key), order\_id (foreign key referencing PURCHASE\_ORDERS), product\_id (foreign key referencing PRODUCTS), quantity, and unit\_price.
   * Links products to purchase orders and tracks quantities ordered.
5. **SALES**
   * Captures sales data: sale\_id (primary key), product\_id (foreign key referencing PRODUCTS), sale\_date, quantity\_sold, and selling\_price.
   * Tracks product sales and revenue.

**Relationships**

* PURCHASE\_ORDERS is linked to SUPPLIERS via supplier\_id.
* PURCHASE\_ORDER\_ITEMS connects PURCHASE\_ORDERS and PRODUCTS through order\_id and product\_id, respectively.
* SALES is linked to PRODUCTS via product\_id.

**Data Population**

The database is populated with sample data to demonstrate functionality:

* **Products**: 4 products (e.g., Wireless Mouse, Keyboard, HDMI Cable, Webcam) with details on price, stock quantity, and reorder levels.
* **Suppliers**: 2 suppliers (Tech Distributors, Hardware Hub) with contact information.
* **Purchase Orders**: 2 orders with total amounts of $10,000 and $5,000.
* **Purchase Order Items**: 3 items across the purchase orders, detailing quantities and unit prices.
* **Sales**: 4 sales transactions, including quantities sold and selling prices.

**Key Features and Queries**

The system supports a variety of queries to facilitate inventory management and decision-making. Below are the key functionalities:

**1. Inventory Monitoring**

* **View All Products**:
  + Query: SELECT \* FROM PRODUCTS;
  + Displays all product details, including name, category, price, stock quantity, and reorder level.
* **Identify Low Stock Products**:
  + Query: SELECT name, stock\_quantity, reorder\_level FROM PRODUCTS WHERE stock\_quantity < reorder\_level;
  + Highlights products with stock levels below their reorder thresholds, enabling timely restocking.
* **Calculate Total Stock Value**:
  + Query: SELECT SUM(price \* stock\_quantity) AS total\_stock\_value FROM PRODUCTS;
  + Computes the total value of inventory, aiding financial assessments.

**2. Supplier Management**

* **List All Suppliers**:
  + Query: SELECT \* FROM SUPPLIERS;
  + Provides a complete list of suppliers and their contact information.

**3. Sales Analysis**

* **Total Units Sold per Product**:
  + Query: SELECT product\_id, SUM(quantity\_sold) AS total\_units\_sold FROM SALES GROUP BY product\_id;
  + Summarizes units sold for each product, useful for demand analysis.
* **Top Selling Products**:
  + Query: SELECT P.name, SUM(S.quantity\_sold) AS total\_sold FROM SALES S JOIN PRODUCTS P ON S.product\_id = P.product\_id GROUP BY P.name ORDER BY total\_sold DESC LIMIT 5;
  + Identifies the top 5 products by units sold, supporting sales strategy.
* **Monthly Sales Revenue**:
  + Query: SELECT DATE\_FORMAT(sale\_date, '%Y-%m') AS sale\_month, SUM(quantity\_sold \* selling\_price) AS total\_revenue FROM SALES GROUP BY sale\_month;
  + Aggregates revenue by month, facilitating financial reporting.
* **Unsold Products**:
  + Query: SELECT name FROM PRODUCTS WHERE product\_id NOT IN (SELECT DISTINCT product\_id FROM SALES);
  + Lists products with no sales, highlighting potential dead stock.

**4. Procurement Tracking**

* **Purchase Order Details**:
  + Query: SELECT PO.order\_id, S.name AS supplier\_name, PO.order\_date, PO.total\_amount FROM PURCHASE\_ORDERS PO JOIN SUPPLIERS S ON PO.supplier\_id = S.supplier\_id;
  + Provides an overview of purchase orders with supplier details.
* **Purchase Order Items Breakdown**:
  + Query: SELECT POI.order\_item\_id, P.name, POI.quantity, POI.unit\_price, (POI.quantity \* POI.unit\_price) AS total\_cost FROM PURCHASE\_ORDER\_ITEMS POI JOIN PRODUCTS P ON POI.product\_id = P.product\_id;
  + Details items in each purchase order, including total cost per item.
* **Purchase Order and Supplier Insights**:
  + Query: SELECT P.name AS product\_name, S.name AS supplier\_name, PO.order\_date FROM PURCHASE\_ORDER\_ITEMS POI JOIN PRODUCTS P ON POI.product\_id = P.product\_id JOIN PURCHASE\_ORDERS PO ON POI.order\_id = PO.order\_id JOIN SUPPLIERS S ON PO.supplier\_id = S.supplier\_id;
  + Connects products, suppliers, and purchase orders for comprehensive procurement tracking.

**5. Sales and Product Integration**

* **Sales Details with Product Names**:
  + Query: SELECT P.name AS product\_name, S.quantity\_sold, S.selling\_price, S.sale\_date FROM SALES S JOIN PRODUCTS P ON S.product\_id = P.product\_id;
  + Combines sales data with product names for detailed sales reporting.

**Implementation Details**

* **Database Creation**: The database is created with CREATE DATABASE InventoryDB; and set as active with USE InventoryDB;.
* **Data Integrity**: Foreign key constraints ensure referential integrity across tables (e.g., supplier\_id in PURCHASE\_ORDERS references SUPPLIERS).
* **Sample Data**: Pre-inserted data simulates real-world scenarios, enabling immediate query testing.
* **Query Optimization**: Queries use appropriate joins, grouping, and filtering to ensure efficient data retrieval.

**Benefits of the System**

* **Efficient Inventory Management**: Tracks stock levels and flags low-stock items to prevent stockouts.
* **Streamlined Procurement**: Manages purchase orders and supplier relationships effectively.
* **Sales Insights**: Provides actionable data on sales performance, top products, and revenue trends.
* **Scalability**: The schema supports additional tables or fields for future enhancements (e.g., customer data, warehouse locations).
* **Data Integrity**: Foreign keys and primary keys ensure consistent and reliable data.

**Future Enhancements**

* **Automated Alerts**: Implement triggers to notify when stock falls below reorder levels.
* **Customer Module**: Add a CUSTOMERS table to track buyer information and link to sales.
* **Reporting Dashboard**: Develop a front-end interface to visualize query results (e.g., stock value, sales trends).
* **Advanced Analytics**: Introduce queries for profitability analysis (e.g., comparing purchase and selling prices).

**Conclusion**

The InventoryDB system provides a robust foundation for managing inventory, procurement, and sales. Its relational structure, combined with targeted queries, supports operational efficiency and data-driven decision-making. The system is ready for immediate use with the provided sample data and can be extended to meet future business needs.