**DataPioneers\_Inventory**

**Team Member:**Anand Somnath Pandey

Kartik Taneja

**Project Topic**

An Inventory Management System using Oracle SQL Developer to efficiently track and manage stock, customers, and orders in a retail environment.

**Problem Statement**

Many small and mid-sized businesses have difficulty keeping track of their inventory. They often face problems like running out of stock, having too much stock, or delays in fulfilling orders. Without a proper system in place, businesses may lose money and struggle to keep customers happy. This project aims to solve these issues by creating a structured database system using Oracle SQL Developer. The system will help businesses track their products, manage customer orders, and improve overall efficiency.

**Objective**

* Develop a structured inventory management system solely using Oracle SQL Developer.
* Implement a well-defined database schema with clear relationships.
* Ensure accurate tracking of products, customer details, and order management.
* Optimize query performance for efficient data retrieval.
* Maintain data integrity using constraints and relationships.

**Database Design Document**

**Business Problem and Solution**

Retail businesses require a structured system for inventory and order management to prevent inefficiencies. Without a proper system, businesses often face challenges such as:

* **Stock Shortages:** Running out of popular products can lead to lost sales and dissatisfied customers.
* **Overstocking:** Holding too much inventory increases storage costs and the risk of product obsolescence.
* **Order Processing Delays:** Without an efficient system, businesses struggle to fulfill orders on time, leading to poor customer satisfaction.
* **Lack of Inventory Visibility:** Many businesses manually track inventory, which results in errors and mismanagement.

The **DataPioneers\_Inventory** system provides a **structured, database-driven approach** to solve these issues by:

* **Implementing a central database** that stores product, customer, and order details, ensuring seamless tracking and management.
* **Utilizing Oracle SQL Developer** to enhance database integrity, minimize errors, and improve efficiency.
* **Automating inventory tracking** to ensure businesses always know stock levels, preventing shortages or overstocking.
* **Optimizing data relationships** for faster and more reliable order processing.
* **Providing real-time insights** into product availability, order history, and supplier details for better decision-making.

By adopting this system, businesses can ensure smoother inventory operations, minimize losses, and enhance customer satisfaction through better order fulfillment.

**Entity Relationship Diagram (ERD)**

**Logical and Physical Models**

The ERD defines the relationships between the key entities in the system.

**Entities and Their Relationships:**

1. **Products** - Stores product details such as name, category, price, and stock levels.
2. **Customers** - Maintains customer details such as name, contact information, and registration date.
3. **Orders** - Captures customer purchases, linking customers and order details.
4. **OrderDetails** - Stores specific product purchases linked to an order.
5. **Suppliers** - Stores supplier details for inventory management.
6. **Warehouse** - Maintains warehouse storage locations for inventory distribution.
7. **ProductWarehouse** - Links products to warehouses, allowing tracking of stock levels.

**Relationships:**

* **Customers place Orders** - One-to-Many (One customer can place multiple orders).
* **Orders contain OrderDetails** - One-to-Many (Each order contains multiple products).
* **OrderDetails reference Products** - Many-to-One (Multiple order details can include the same product).
* **Products are supplied by Suppliers** - Many-to-One (A product has one supplier, but a supplier can provide multiple products).
* **Products are stored in Warehouses through ProductWarehouse** - Many-to-Many (A product can be stored in multiple warehouses, and a warehouse can store multiple products).

**Entity and Attributes with Defined Data Types**

**Products**

* ProductID (NUMBER, Primary Key)
* Name (VARCHAR2(255), NOT NULL)
* Category (VARCHAR2(100))
* Price (NUMBER(10,2), NOT NULL)
* StockQuantity (NUMBER, NOT NULL)
* SupplierID (NUMBER, Foreign Key references Suppliers(SupplierID))
* CreatedAt (TIMESTAMP DEFAULT CURRENT\_TIMESTAMP)
* UpdatedAt (TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP)

**Customers**

* CustomerID (NUMBER, Primary Key)
* FirstName (VARCHAR2(100), NOT NULL)
* LastName (VARCHAR2(100), NOT NULL)
* Email (VARCHAR2(255), UNIQUE, NOT NULL)
* Phone (VARCHAR2(20))
* CreatedAt (TIMESTAMP DEFAULT CURRENT\_TIMESTAMP)

**Orders**

* OrderID (NUMBER, Primary Key)
* CustomerID (NUMBER, Foreign Key references Customers(CustomerID))
* OrderDate (TIMESTAMP DEFAULT CURRENT\_TIMESTAMP)
* TotalAmount (NUMBER(10,2), NOT NULL)

**OrderDetails**

* OrderDetailID (NUMBER, Primary Key)
* OrderID (NUMBER, Foreign Key references Orders(OrderID))
* ProductID (NUMBER, Foreign Key references Products(ProductID))
* Quantity (NUMBER, NOT NULL)
* SubTotal (NUMBER(10,2), NOT NULL)

**Suppliers**

* SupplierID (NUMBER, Primary Key)
* CompanyName (VARCHAR2(255), NOT NULL)
* ContactPerson (VARCHAR2(100))
* Phone (VARCHAR2(20))

**Warehouse**

* WarehouseID (NUMBER, Primary Key)
* Location (VARCHAR2(255), NOT NULL)

**ProductWarehouse**

* ProductID (NUMBER, Foreign Key references Products(ProductID))
* WarehouseID (NUMBER, Foreign Key references Warehouse(WarehouseID))
* StockLevel (NUMBER, NOT NULL)
* LastUpdated (TIMESTAMP DEFAULT CURRENT\_TIMESTAMP)
* **Primary Key: (ProductID, WarehouseID)**

**ERD:**

A diagram of a data flow

AI-generated content may be incorrect.