**Inventory Management System for a Store**

**Author:** Shyaka Bosco Domin  
**Student ID:** 27406  
**PDB Name:** E\_27406\_SHYAKA\_INVENTORY\_DB  
**Course: Database Development with PL/SQL (INSY 8311)  
Instructor:** Eric Maniraguha

**Overview**

This project implements an Oracle PL/SQL-based Inventory Management System for a retail store. It automates stock control, order processing, sales recording, and auditing with advanced features such as triggers and packages.

**Phase I: Problem Statement & Goals**

**Simple Problem:** A store frequently runs out of fresh items due to poor manual tracking and lack of reorder alerts.  
**Problem:** Manual inventory tracking causes stock mismanagement, spoilage of fresh foods, and financial loss.  
**Context:** Retail stores and warehouses.  
**Target Users**: Store managers, employees, and suppliers.  
**Goals:**

* Automate inventory tracking
* Reduce stockouts and overstocking
* Improve decision-making
* Enhance security using role-based access

**Phase II: Business Process Modeling (MIS)**

**Simple Problem:** Employees sell products without knowing when to restock, and managers receive no timely alerts.

**Tool Used:** draw.io  
**Actors:** Employee, System, Manager, Supplier  
**Swimlane Process:**

1. Employee records sale
2. System updates inventory
3. System checks threshold
4. If low stock, Manager is alerted
5. Manager approves order
6. Supplier delivers items
7. System updates stock

**Diagram*:*** *phase2\_inventory\_process\_diagram.drawio PNG export***Text Explanation:** Provided in description.txt

**Phase III: Logical Model Design (ERD)**

**Simple Problem:** There is no centralized structure linking products, sales, users, and suppliers, making data tracking difficult.

**Entities:** Products, Suppliers, Orders, Users, Sales, Holidays  
**Normalization:** All tables are in 3NF  
**Relationships:**

* Products <-> Orders <-> Suppliers
* Products <-> Sales <-> Users
* Holidays used for restricted operations

**Tools:** draw.io (ERD file: er\_diagram.drawio)  
File: normalization\_notes.txt summarizes normalization approach

**Phase IV: Database Creation**

**Simple Problem:** Data was previously stored in Excel sheets with no secure multi-user access.

User Created: shyaka\_user with password Bosco  
Privileges: CONNECT, RESOURCE, UNLIMITED TABLESPACE  
Pluggable Database: E\_27406\_SHYAKA\_INVENTORY\_DB

All tables were created under shyaka\_user using create\_tables.sql

**Phase V: Data Insertion**

Simple Problem: Test data is required to simulate store operations such as sales,

Sample Data Added For:

* Products
* Suppliers
* Users
* Orders, Sales, Holidays

**Phase VI: Procedures, Transactions, Packages**

**Simple Problem:** Repetitive SQL operations like adding sales or updating inventory are prone to human error without automation.

**Focus:** Modularize logic, fetch data, handle exceptions

**What was done:**

* Created reusable PL/SQL packages (audit\_pkg) for audit logging
* Applied DML operations using cursors and exception blocks
* Sample procedures created for inserting and updating inventory

**Phase VII: Advanced Programming + Auditing**

**Simple Problem:** Employees sometimes alter sales records during work hours or holidays without approval, risking data integrity.

**Security Enhancements:**

* Restricted Employees from modifying data on weekdays (Mon-Fri) and public holidays
* Used the holidays table to identify restricted days

**Auditing:**

* All operations log to audit\_log table via audit\_pkg
* Each record stores user\_id, timestamp, operation, status, message

Files:

* audit\_pkg.sql: Logging package
* trg\_restrict\_employee\_ops.sql: Restriction + audit trigger
* audit\_log.sql: Audit log table definition