DataPioneers Inventory Management System

This project demonstrates a normalized SQL-based inventory system with support for order placement, stock tracking, and automatic fulfillment of backorders. It is implemented using Oracle SQL and follows best practices in schema design, constraints, business rules, and modular script execution.

GitHub Repo Link: https://github.com/KartikeTaneja/DataPioneers

Execution Flow

All SQL scripts are stored in the DataPioneers_Scripts/ directory.

Always connect and run scripts as data_pioneers_admin (admin)

Step 1: System Setup

1.1 User Creation and Privileges

Script: DataPioneers_SetUp.sql

This script creates the two users `data_pioneers_admin` and `data_pioneerss`:

- data_pioneers_admin` is the admin user with full privileges.
- data_pioneers` is the regular user with limited privileges(read-only access).

1.2 **Execution Steps:**

1. Connect to Oracle as a user with SYSDBA privileges
2. Run the setup script: DataPoineers_SetUp.sql

Step 2: Schema Creation

2.1 Database Schema

Script: DataPioneers_Validated_Schema.sql

This script creates the complete database schema including:

- Tables with appropriate constraints
- Views for business reporting
- Performance optimization indexes

Execution Steps:

1. Connect as data_pioneers_admin

2. Run the schema script: DataPioneers_Validated_Schema.sql
3. Verify tables and views created

Step 3: Data Population

3.1 Sample Data Insertion

Script: DataPioneers_SampleData.sql

This script populates all tables with sample data including customers, suppliers, products, orders, and warehouses. It also ensures all views work with the sample data.

Execution Steps:

1. Ensure you're connected as data_pioneers_admin
2. Run the sample data script: DataPioneers_SampleData.sql

Step 4: Business Logic Implementation

4.1 Functions and Packages

Script: inventory_function_and_package.sql

This script creates business logic components:

Function: get_inventory_value(p_product_id)

Procedure: fulfill_backorders(p_product_id)

Package: inventory_pkg containing both the function and procedure

Execution Steps:

1. Ensure you're connected as data_pioneers_admin
2. Run the package script: inventory_function_and_package.sql

Step 5: Testing the System

5.1 Backorder Fulfillment Test

Script: DataPioneers_Test_Backorder.sql

This script tests the backorder fulfillment functionality by:

1. Replenishing stock for a product
2. Running the fulfillment procedure
3. Verifying the order status changes and stock quantities update correctly

Execution Steps:

1. Ensure you're connected as data_pioneers_admin
2. Run the test script: DataPioneers_Test_Backorder.sql
3. Verify the test results in the output

5.2 Read-Only User Testing

Execution Steps:

1. Connect as the read-only user: data_pioneers
2. Access objects using fully qualified names

Step 6: System Verification

6.1 Full System Check

After all scripts are executed, verify the complete system:

1. Connect as admin
2. Check all tables have data
3. Verify views return data
4. Check backorder status

Step 7: System Features

Key System Capabilities

- **Normalized Schema Design**: All tables conform to 1NF, 2NF, and 3NF
- **Business Rules Implementation**: Via constraints and triggers
- **Security Model**: Separate admin and read-only users
- **Reporting**: Business intelligence views
- **Performance Optimization**: Strategic indexing
- **Modular Code**: Functions and packages for business logic

Business Rules Enforced

- Product quantities must be non-negative
- Prices must be positive and reasonable (0-10,000)
- Email addresses must be unique
- Phone numbers must be valid (at least 10 digits)
- Orders have defined statuses (Pending, Fulfilled, Backordered)